Mobile Information Systems

Personalized Distributed Machine Learning for Mobile Services

Lead Guest Editor: Yan Huang
Guest Editors: Madhuri Siddula and Xu Zheng

Personalized Distributed Machine Learning for Mobile Services

Mobile Information Systems

Personalized Distributed Machine Learning for Mobile Services

Lead Guest Editor: Yan Huang
Guest Editors: Madhuri Siddula and Xu Zheng

Contents

Hindawi
Mobile Information Systems
Volume 2022, Article ID 8966266, 9 pages
https://doi.org/10.1155/2022/8966266

Research Article

Explainable and Personalized Medical Cost Prediction Based on Multitask Learning over Mobile Devices

Lin Sun,[1,2] Tingqi Wang,[1] Bei Hui,[3,4] Yun Li,[5] and Ling Tian[1,6]

[1]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
[2]*West China Hospital of Sichuan University, Chengdu 610000, China*
[3]*School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610031, China*
[4]*Kashi Institute of Electronic and Information Industry, Kashi 844000, China*
[5]*Department of Thoracic Surgery, The Seventh Affiliated Hospital, Sun Yat-Sen University, Shenzhen 528406, China*
[6]*Shenzhen Institute of Information Technology, Shenzhen 518172, China*

Correspondence should be addressed to Bei Hui; bhui@uestc.edu.cn

Received 15 August 2022; Accepted 6 September 2022; Published 9 October 2022

Academic Editor: Yan Huang

Currently, the forecasting of healthcare costs is of significant importance for the finance management of both government and individual citizens. However, the existence of dramatic individual diversity in health status, as well as the extensive complexity of the factors influencing the cost, has made the prediction a challenging task. Thanks to the unprecedented adoption of mobile devices, regular individuals may contribute diverse dimensions of data for the medical cost prediction. Hospitals and healthcare service providers are all setting up their own mobile services and collect user data for analysis. Previous methods usually employed traditional machine learning or simple neural network methods, which are difficult to be applied to the nonlinear medical cost and diverse dimensions of data. Therefore, this paper proposes a multitask learning-based framework for interpretable medical cost interval prediction to address these issues. The framework proposed in this paper first predicts subcost intervals by applying the multidimensional data collected from mobile ends and following the multitask learning paradigm. The total cost interval is then predicted based on this prediction. Simultaneously, the framework derives a decision tree from the parameters of the multitask learning network and calculates the importance of each feature in predicting the cost intervals. This paper demonstrates the method's effectiveness using real-world data experiments.

1. Introduction

The management of healthcare cost is one of the largest challenges in the field of health insurance and healthcare, which can easily lead to a shortage or waste of healthcare resources when poorly managed [1–4]. Owing to the extensive development of mobile devices, patients and regular citizens can freely contribute their own data for the prediction of the medical cost. Typical organizations like healthcare service providers and hospitals are setting up their own applications towards this trend. Patients and subscribers can use those mobile apps to contribute multiple types of data like demographic attributes, manually inputted daily healthcare records, and even sensing data from smart watches [5]. Therefore, it is of great significance to study the adoption of these data for medical cost prediction, which can bring personalized and understandable services for patients.

Currently, DRG (Diagnosis Related Group)-based payment methods are being widely used to predict costs through characteristic groupings [6, 7], which has strongly motivated the research on reliable medical cost prediction. Various methods are proposed to accurately predict cost ranges and identify key factors for grouping, allowing for efficient resource deployment and timely identification of potential risks. These methods are assumed to bring significant implications for reducing pressure on healthcare resources and improving resource utilization [8, 9] while concealing no significant personal information of patients

[10–12]. However, because of the various treatment options chosen by individual patients, the amount and composition of medical cost are highly personalized and divergent [13]. Moreover, due to the different conditions of different patients and the influence of factors such as healing time and degree of recovery, it is difficult to fit medical costs with simple linear models. Therefore, the prediction of medical cost requests both the application of various dimensions of data available from mobile ends and feedbacks to users with a deep understanding on the impact of individual and personalized characteristics on healthcare costs [14, 15].

Considering these challenges, traditional methods rely heavily on machine learning models like linear regression [16] and regression trees [17, 18], as well as simple neural network models. However, the overall representation is inadequate owing to the sophisticated correlations among factors. In recent research, deep learning methods [19] outperformed traditional computational methods in various prediction tasks due to their ability to adapt the composition of individual feature factors for better representation [20]. Given the complexity of the components and data dimensions in medical cost prediction [21], deep learning methods can make more accurate and reasonable predictions of overall costs by depicting the correlation between the various costs in addition to predicting individual costs.

Based on the above, this paper proposes a multi-task learning-based interpretable medical cost interval prediction framework. The model takes multiple sources of information about the patient into account, including (1) the patient's natural characteristics and (2) the stage of the patient's condition. (3) The patient's lesion attributes, and outputs the prediction results for each type of cost interval.

The framework is made up of two parts: (1) A multitask-learning framework for interval prediction over data collected from mobile ends. The cost intervals are predicted by the prediction framework in two steps. To begin, a logistic regression approach is combined as a preprocessing of the input neural network data, which is then fed into the neural network to calculate predictions for the various subcost intervals. The total costs are then predicted based on the prediction of the subcost intervals. Among these, the logistic regression method is used to improve the network's convergence and training speeds. (2) An explainable and personalized decision tree based on the analysis of factor importance in a multilearning task. The Gini coefficient is reconstructed using the multitask learning framework weights obtained from training to build a decision tree, and the importance of each feature is calculated using the decision tree.

The proposed framework owns two advantages for medical cost prediction. On the one hand, the framework predicts total costs by coupling subcosts, allowing all subcost prediction intervals to be in obtained while also capturing the links between subcosts and global payments; on the other hand, the framework can analyze the importance of different factors in the prediction of cost intervals based on the prediction process. Corresponding observations can serve as a foundation for physician triage.

To the best of our knowledge, this is the first time that a multiclassification approach to cost interval forecasting has been used. The remainder of this paper is organized as follows: Section 2 presents work related to cost interval prediction. Section 3 presents the cost interval prediction model for multitask learning. Section 4 presents the experimental results. Section 5 analyses the factor impact. Section 6 presents the conclusions.

2. Related Work

The study of cost prediction tasks is becoming more widespread, and one of the widely used methods for health care cost prediction is the regression-based model [22, 23]. To avoid the requirement of general linear models for data to follow a normal distribution, Moran et al. performed prediction using generalized linear models [16]. Panay et al. used the evidence regression method, which is based on the idea that other elements in a set that are correlated for a specific element are placed in a set of similar patients, and the overall predicted expectation is calculated for optimization [24]. Tkachenko R et al. used SGTM-like neural structures for segmented linear prediction [25]. Takeshima et al. defined experimental valuables on which regression models with minimum absolute shrinkage and selection operators (lasso) were built. Explanatory valuables were selected by LASSO avoiding overfitting using the validation data [26]. Based on regression methods, various machine learning methods have been introduced [27, 28]. Taloba et al. in [17] compare the performance of linear regression type Lasso, gradient augmentation of regression decision trees, M5 regression decision trees, random forests, linear regression, and CART regression trees in this task and analyze the advantages and disadvantages of each method.

Due to properties such as end-to-end training and good fitting ability to nonlinear data, neural network methods, in addition to machine learning methods, have been introduced into the prediction of medical costs. Morid et al. compared various methods and found that ANN (Artificial Neural Network) performed the best [20]. In [29], Zeng et al. used multilayer neural networks to construct unsupervised learning models to learn patient representation from medical data. The collection of medical data from mobile devices are also extensively studied. Issues like efficiency [5, 30] and data utilities are thoroughly considered. These studies are complementary to our work.

Generally, for cost prediction, current work is primarily based on patients' natural attributes and health data, but there are fewer methods for predicting the costs of specific conditions during treatment. At the same time, current methods are based on simple statistical learning and neural networks, and they are incapable of fully exploiting the value of data contributed by patients from mobile ends.

3. Framework: A Multitask Learning Based Framework for Interpretable Medical Cost Interval Prediction

3.1. Problem Definition. For a patient set $U = \{U_1, U_2, ..., U_i, ..., U_N\}$ containing N patients, where each patient U_i has a feature set X_i and an element $x_i \in R$ for each feature

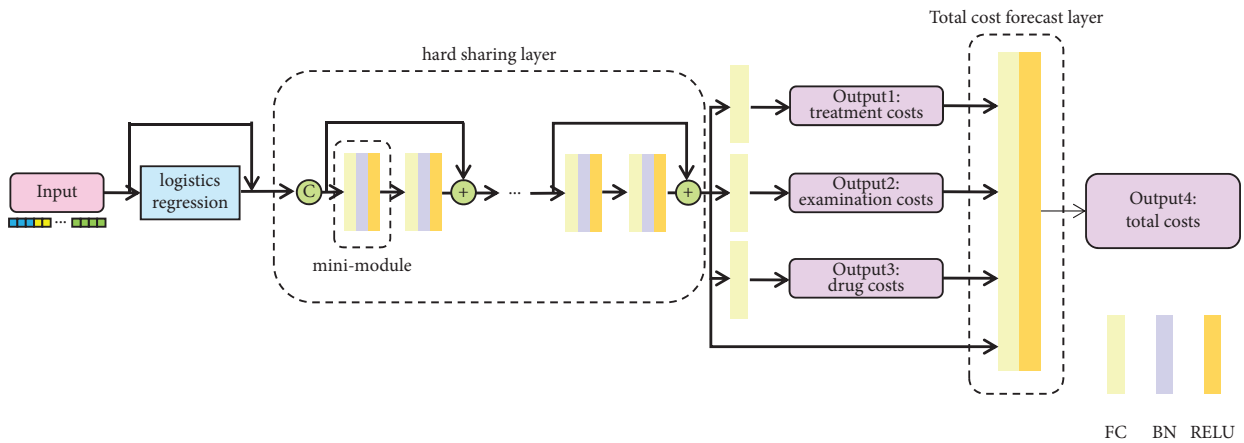


Figure 1: The cost forecasting framework.

dimension. For example, the feature set may include natural features such as "age," daily collected data like heartbeat records and related events inputted by patients. These features are collected and submitted through mobile devices. Corresponding features also involve the disease stage such as "TMN-stage," and focal features such as "type of comorbidity," which are both terminologies used for clinic diagnosis of breast cancer. We define a cost interval set $Y = \{Y_1, Y_2, \ldots, Y_i, \ldots, Y_N\}$, where $Y_i = Y_{i,\text{sub}} \cup Y_{i,\text{total}}$, $Y_{i,\text{sub}} = \{y_{i,1}, y_{i,2}, \ldots, y_{i,k}\}$, $Y_{i,\text{total}} = \{y_{i,\text{total}}\}$.

In this paper, we take $k = 3$ as an example, $y_{i,1}, y_{i,2}, y_{i,3}$ correspond to the intervals of treatment cost, examination cost, and drug cost, respectively, and $y_{i,\text{total}}$ is the total cost interval. $y_{i,1} \in \{1, 2, \ldots, k_1\}$, $y_{i,2} \in \{1, 2, \ldots, k_2\}$, $y_{i,3} \in \{1, 2, \ldots, k_3\}$, $y_{i,\text{total}} \in \{1, 2, \ldots, k_{\text{total}}\}$.

For a given set of patient features X, after inputting it into the model, the set $Y = Y_{\text{sub}} \cup Y_{\text{total}}$ of its corresponding cost intervals is output.

3.2. A Framework for Predicting Medical Cost Intervals Based on Multitask Learning. The framework proposed in this paper consists of three components: data preprocessing; a hard sharing network for subcost interval prediction; and a total task prediction network based on sub-cost intervals. The results obtained from predicting subcost intervals and the raw data outputted by hard sharing are used as inputs for the total cost prediction. An illustration of the framework is shown in Figure 1.

3.2.1. Data Preprocessing: Logistic Regression. The training of neural networks for such data suffers from slow convergence and long training times due to the weak linear nature of the association between medical data and medical cost intervals. Traditional machine learning methods like logistic regression may extract shallow nonlinear association among data, which can benefit the overall training performance of the framework. As a result, in this paper, $\hat{\theta}_{\text{log istic}}$ is calculated using logistic regression before being fitted with a neural network. When the user U_i set of features X_i is entered, the auxiliary information $h_{\hat{\theta}_{\text{log istic}}}(X_i)$ can be obtained.

$$h_{\hat{\theta}_{\text{log istic}}}(X_i) = \text{softmax}\left(X_i; \hat{\theta}_{\text{log istic}}\right) = \begin{bmatrix} p\left(y_{i,\text{total}} = 1 | X_i; \hat{\theta}_{\text{log istic}}\right) \\ p\left(y_{i,\text{total}} = 2 | X_i; \hat{\theta}_{\text{log istic}}\right) \\ \cdot \\ p\left(y_{i,\text{total}} = k_{\text{total}} | X_i; \hat{\theta}_{\text{log istic}}\right) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\hat{\theta}_j^T X_i}} \begin{bmatrix} e^{\hat{\theta}_1^T X_i} \\ e^{\hat{\theta}_2^T X_i} \\ \cdot \\ e^{\hat{\theta}_{k_{\text{total}}}^T X_i} \end{bmatrix}, \quad (1)$$

where $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_{k_{\text{total}}} \in R$ is the model parameter and the parameter is obtained by optimising the loss function by an iterative method, the loss function is a great likelihood function, $\hat{\theta}_{\text{log istic}} = \underset{\theta_{\text{log istic}}}{\text{argmin}}\, \text{Loss}_{\text{log istic}}(h_{\theta_{\text{log istic}}}(X), Y)$.

$h_{\hat{\theta}_{\text{log istic}}}(X_i)$ concatenated with the original data, to obtain $\text{INPUT}(X_i)$:

$$\text{INPUT}(X_i) = \text{concat}\left(X_i, h_{\hat{\theta}_{\text{log istic}}}(X_i)\right). \quad (2)$$

$\text{INPUT}(X_i)$ as input to the multitask learning hard sharing layer can speed up the convergence and training of the neural network.

3.2.2. Hard Sharing Network for Subcost Interval Forecasting. The hard sharing layer consists of the mini-module and Resnet.

Each mini-model consists of a full connection layer, a BatchNormalization layer, and an activation layer (ReLu is used as an example in this paper) which, after the hard sharing layer, gives a hidden layer representation of the data $m_i^{(l)}$:

$$\begin{aligned} FC_i^{(l)} &= FC\left(m_i^{(l-1)}\right), \\ BN_i^{(l)} &= BN\left(FC_i^{(l)}\right), \\ m_i^{(l)} &= \sigma\left(BN_i^{(l)}\right), \end{aligned} \tag{3}$$

where l is the number of layers in the network of the mini-module.

In order not to degrade the performance of the network due to degradation caused by nonconstant mapping, a residual network is used in this paper. A residual connection [19] is made for every two mini-modules to obtain the hidden layer $h(X_i)$:

$$\begin{aligned} \text{Resnet}_i^{(L)} &= \text{Resnet}\left(m_i^{(2L)}, m_i^{(2L+1)}\right), \\ h(X_i) &= \text{Resnet}_i^{(L)}. \end{aligned} \tag{4}$$

Put $h(X_i)$ into different full connection layers to obtain predictions for each subcost interval:

$$\begin{aligned} \hat{y}_{i,1} &= FC_1\left(h(X_i)\right), \\ \hat{y}_{i,2} &= FC_2\left(h(X_i)\right), \\ \hat{y}_{i,3} &= FC_3\left(h(X_i)\right). \end{aligned} \tag{5}$$

Based on this, $\hat{Y}_{i,\text{sub}} = \left\{\hat{y}_{i,1}, \hat{y}_{i,2}, \hat{y}_{i,3}\right\}$ of X_i is obtained. Where the loss function Loss_{sub} for the subcost prediction network is defined as follows:

$$\text{Loss}_{\text{sub}} = \sum_{j=1}^{3} \alpha_j \text{Loss}\left(\hat{Y}_{i,j}, Y_{i,j}\right). \tag{6}$$

3.2.3. Total Cost Interval Forecasting Network Based on Subcost Intervals. A fully connected layer and an activation layer comprise the total cost interval prediction network. The predicted values of the three subcost intervals, along with the output of the hard sharing layer, are fed into the total cost prediction layer, which produces a prediction of the total cost interval as follows:

$$\begin{aligned} \hat{X}_{i,\text{total}} &= \text{concat}\left(h(X_i), \hat{y}_1, \hat{y}_2, \hat{y}_3\right), \\ \hat{y}_{i,\text{total}} &= \sigma\left(FC\left(\hat{X}_{i,\text{total}}\right)\right), \end{aligned} \tag{7}$$

From this, the predicted value $\hat{Y}_i = \left\{\hat{y}_{i,\text{total}}\right\}$ is obtained for four cost intervals of X_i

The loss function $\text{Loss}_{\text{total}}$ for the overall cost is defined as follows:

$$\text{Loss}_{\text{total}} = \text{Loss}\left(\hat{Y}, Y\right). \tag{8}$$

It distinguish the differences between Loss_{sub} and $\text{Loss}_{\text{total}}$:

$$\text{Loss} = \beta_1 \text{Loss}_{\text{sub}} + \beta_2 \text{Loss}_{\text{total}}, \tag{9}$$

where β_1 and β_2 are hyper parameters. In this paper, (8) is selected as the loss function.

4. Feature Importance Analysis

Simply predicting the cost interval may confuse the doctors even if a highly accurate performance is guaranteed. Therefore, a feature-importance-based framework in explaining the prediction of cost interval is further proposed in this part. The whole framework is based on an improved version of decision tree, where multiple factors considered in the prediction model are involved. An illustration is shown in Figure 2.

A decision tree approach is used in this paper to analyze the importance of factors obtained through the multitask neural network in section 3. In contrast to previous decision tree methods simply estimating information gain for a single task, an method tailored to couple with multitasks is designed for the information gain estimation.

Based on the weight parameters of the whole prediction network obtained from training, the weight parameters corresponding to each sub-cost interval is first calculated as a percentage of the total cost prediction layer, which is used as the weight for the Gini coefficient calculation of the decision tree nodes. The original Gini coefficient calculation formula:

$$\text{gini} = \sum_{k=1}^{k} p_k\left(1 - p_k\right). \tag{10}$$

For a given matrix of patient features $(x_{1,0}, x_{1,1}, x_{2,0}, x_{2,1}, x_{2,2})$, the input total cost prediction layer is subject to the following calculation: $(x_{1,0}, x_{1,1}, x_{2,0}, x_{2,1}, x_{2,2}) \times (w_{1,0}, w_{1,1}, w_{2,0}, w_{2,1}, w_{2,2})^T$, where the feature elements with the same first numerical ordinal number of the subscript, e.g., $x_{1,0}, x_{1,1}$ refer to common features. Then, for the same feature element X_i, having a matrix of weights W_i, the weight of each feature element in the calculation of the Gini coefficient is calculated as follows:

$$\alpha_k = \frac{\left\|W_k\right\|_{l_2}}{\sum_{i=1}^{N}\left\|W_i\right\|_{l_2}}, \tag{11}$$

Then, the weighted Gini coefficient calculation formula:

$$\begin{aligned} \text{gini} = \alpha_1 \sum_{K_1=1}^{K_1} p_{k_1}\left(1 - p_{k_1}\right) + \alpha_2 \sum_{k_2=1}^{K_2} p_{k_2}\left(1 - p_{k_2}\right) + \ldots \\ + \alpha_n \sum_{k_n=1}^{K_n} p_{k_n}\left(1 - p_{k_n}\right). \end{aligned} \tag{12}$$

We build a decision tree using the CART classification tree method [20]. The main idea of the method is to

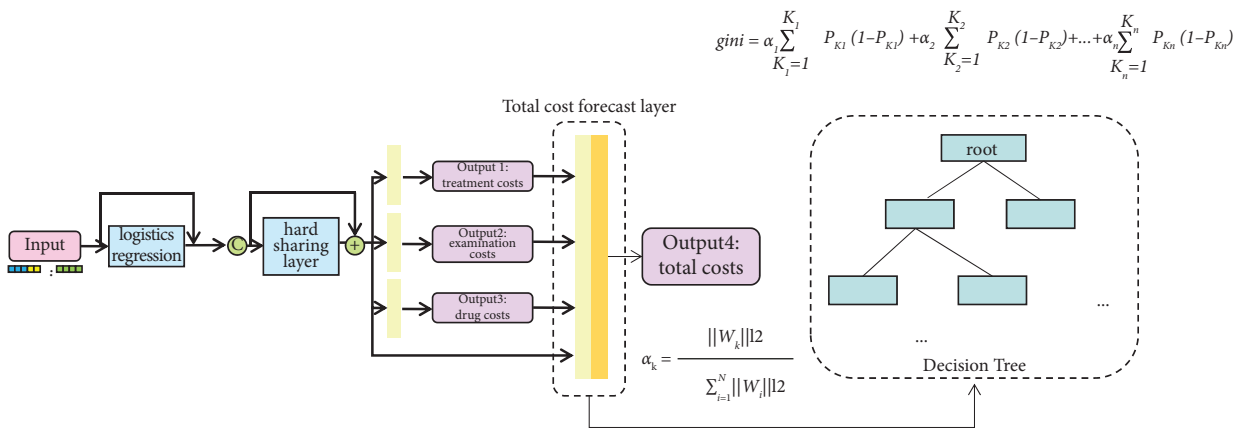


Figure 2: Decision tree analysis method based on neural network weights.

iteratively split the patient set where each subset share identical value on some features. Specifically, when a feature F takes the value f in a sample U with N users, the sample U is divided into two parts U_1 and U_2, where U_1 is the set of samples with $F \neq f$ and U_2 is the set of samples with $F \neq f$. The method calculates the Gini coefficient of each feature at each value, choose the case with the smallest Gini coefficient, and use it to generate this node, with U_1 and U_2 as patient sets in two child nodes. When a node's number of samples U falls below a predefined threshold, or when the number of features is zero, the current node's decision making process is terminated.

The method described above is used to create a decision tree. The essence is to create a binary tree by selecting the features that will give the greatest Gini gain as nodes at each layer. The importance of each feature in nodes is calculated using the following formula based on the generated decision tree:

$$\frac{N_t}{N} \times \left(\text{gini} - \frac{N_{t_R}}{N_t} \times \text{gini}_R - \frac{N_{t_L}}{N_t} \times \text{gini}_L \right), \tag{13}$$

where N is the total number of samples, N_t is the number of samples at this node, N_r is the number of samples at the right child node, and N_{t_L} is the number of samples at the left child node.

5. Experiments

5.1. Dataset. The experiments in this paper are based on a real breast cancer medical cost dataset. We give links to the data demos at the end of this article. Patient features include age, T stage, M stage, N stage, histological classification, complication and comorbidity, and HER2 attributes. The representation for the feature is shown in Table 1.

In this paper, treatment costs, examination costs, and drug costs are selected as the three subcosts to be predicted and the total costs are predicted by using these three subcosts as auxiliary information. For the different costs, the paper divides the cost intervals as shown in Table 2.

The experiments in this paper use one-hot coding for the representation of the data.

Table 1: Patient feature categories classification.

Features	Feature categories	Values
Age	25 -	0
	25–40	1
	40–50	2
	50–60	3
	60+	4
T-stage	T0	0
	Tis	1
	Tx	2
	T1	3
	T2	4
	T3	5
	T4	6
Histologic classification	Invasive carcinoma	0
	Invasive ductal carcinoma	1
	Papillary carcinoma	2
	Infiltrating lobular carcinoma	3
	Medullary carcinoma	4
N-stage	N0	0
	N1	1
	N2	2
	N3	3
	Nx	4
M-stage	M0	0
	M1	1
	Mx	2
Complication and comorbidity	serious	0
	General	1
	—	2
HER2	0	0
	1+	1
	2+	2
	3+	3

5.2. Parameter Settings. The logistic regression model in this paper employs Newton's method as the optimization method for the loss function, and the regularization method employs the l_2 norm with a regularization strength of 0.5; the neural network employs ASGD as the optimization method, with a starting learning rate of 0.1 and decreasing to 50% of the original every 100 epochs; and the linear layer has a dimension of 128.

Table 2: Cost interval.

Treatment costs	1000–	1000–2000	2000+	—
Examination costs	5000–	5000–10000	10000+	—
Drug costs	500–	500–1500	1500–2000	2000+
Total costs	10000–	10000–20000	20000–25000	25000+

Table 3: Accuracy comparison with classical methods.

Method	Accuracy			
	Treatment costs	Examination costs	Drug costs	Total costs
SVM	0.54	0.48	0.53	0.43
DecisionTreeClassifier	0.61	0.54	0.54	0.45
Naive_Bayes	0.59	0.58	0.61	**0.61**
Logistic regression	**0.65**	**0.64**	**0.62**	0.51
k-means	0.48	0.43	0.37	0.42
Multi-task	**0.7**	**0.73**	**0.72**	**0.71**

The decision tree for analyzing the importance of the influencing factors in this paper uses a CART decision tree with a maximum number of layers of 7. The Gini coefficient is used to calculate the information gain, but unlike the traditional Gini coefficient, the Gini coefficient is improved in this paper, and the specific method is described in Section 4.

5.3. Experimental Results

5.3.1. Cost Interval Forecast Results. First, to verify the effectiveness of the methods, SVM, decision tree, plain Bayesian, logistic regression, and k-means methods were tested on the same dataset in this paper. The results are shown in Table 3. Compared with traditional machine learning methods, the multi-task learning method has significantly improved the prediction accuracy for the four types of cost intervals, which raises the accuracy by 5% on treatment cost, 9% on examination cost, 10% on drug cost, and 10% on the total cost.

It can also be seen that the prediction accuracy of our method is also significantly improved compared to that of the logistic regression-only method. The multitask learning model can effectively reduce the reliance on the linear nature of the data using the logistic regression-only method. According to Figure 3, the model convergence speed is improved after the inclusion of the logistic regression approach.

To verify the effectiveness of the framework in this paper, we test the prediction results when the network layer in the framework is replaced by traditional machine learning methods. Moreover, the accuracy of the prediction of subcosts is tested under different total cost prediction results. According to the results in Table 4, when the total cost prediction is correct, the method fails in all cases for the subcost intervals only 21% of the time, which is much lower than traditional machine learning methods. Correspondingly, according to the results in Table 4, when the total cost prediction is incorrect, the sub-cost interval prediction fails

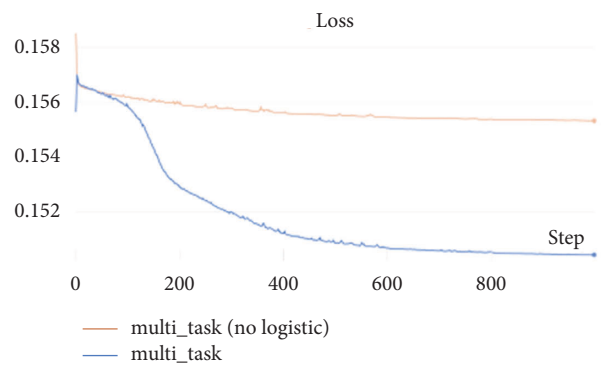


Figure 3: Comparison of loss before and after using the logistic regression method.

in all cases by 24% compared to the correct case, which is the largest improvement compared to the other cases and remains lower than the traditional method. Thus, it can be demonstrated that the neural network method used in this paper, which can better capture the non-linear relationship between subcosts and total costs, outperforms traditional machine learning methods.

Finally, to verify the robustness of the framework, the operation of the network is tested at different learning rates in this paper, and the results are shown in Figure 4. The convergence rate is fast at higher learning rates, but the accuracy as well as the loss gradually converge to the same level at the end. This proves that the network is stable.

5.3.2. Experimental Results on the Feature Importance. Decision trees based on the trained network are shown in Figure 5. Compared with the decision trees built by the traditional method, the prediction accuracy of the our decision tree for the total cost is 0.71, which is much greater than the 0.45 of the traditional decision tree method. The decision tree generated by this method has more Gini nodes with 0 and a clearer judgement process.

Table 4: Percentage of cost forecast (1) Table of other cost projections when total cost projections are correct. (2) Table of other cost projections when total cost projections are incorrect.

Methods	Accuracy	
	Number of correct projections for other costs ≥ 1	Number of correct projections for other costs = 0
SVM	0.54	0.45
DecisionTreeClassifier	0.62	0.38
Naive_Bayes	0.51	0.49
Logistic	0.64	0.36
k-means	0.53	0.47
Multi-task	0.79	0.21
	Accuracy	
Methods	Number of correct projections for other costs ≥ 1	Number of correct projections for other costs = 0
SVM	0.47	0.53
DecisionTreeClassifier	0.51	0.49
Naive_Bayes	0.48	0.52
Logistic	0.41	0.59
k-means	0.49	0.51
Multi-task	0.55	0.45

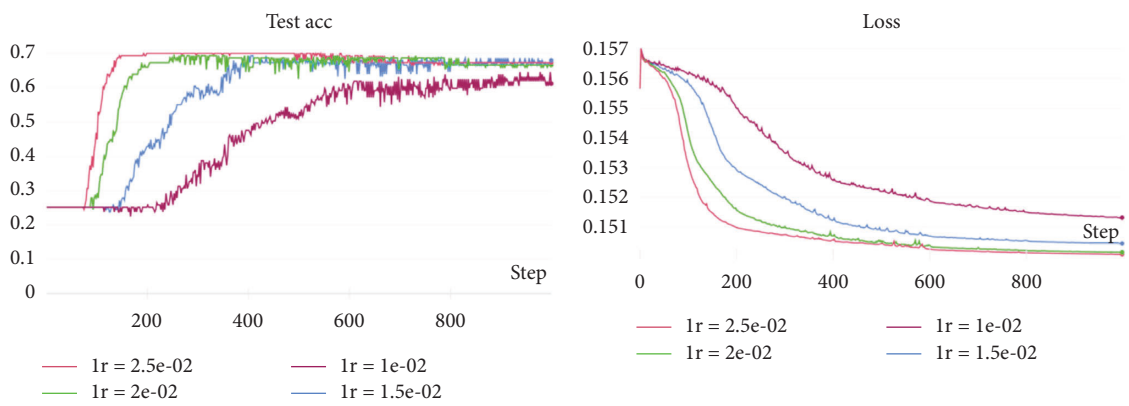


Figure 4: Accuracy and loss of training at different learning rates. (a) Accuracy. (b) Loss.

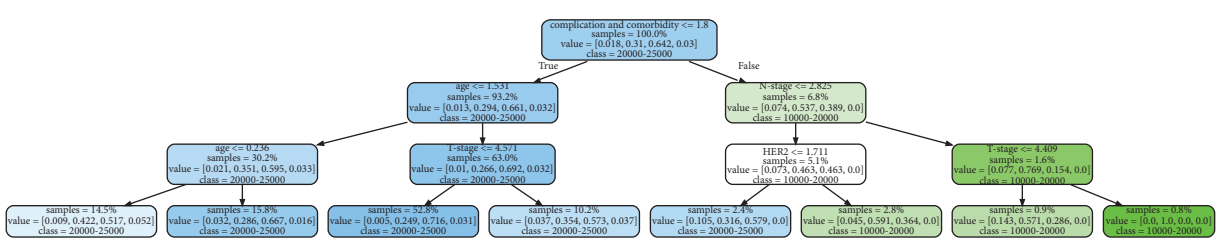


Figure 5: Illustration of the decision tree constructed by our framework.

Table 5: Importance of each feature.

Feature	Importance (descending order)
N-stage	0.1934
T-stage	0.1747
Age	0.1344
Complication and comorbidity	0.1276
HER2	0.1033
Histologic classification	0.0997
M-stage	0.0701

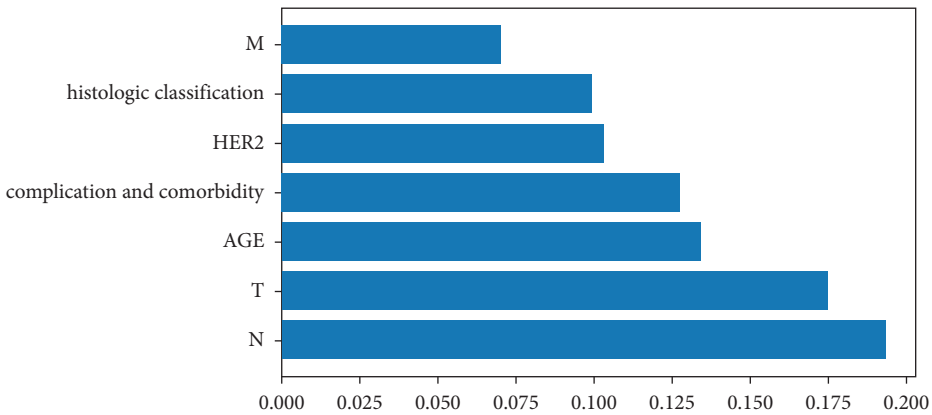


Figure 6: Importance of features.

Based on the generated decision tree, the importance of the features is calculated and the results are shown in Table 5 and Figure 6. Among them, N-stage and T-stage are significantly more important than the last five features, and M-stage is significantly less important than the first six features.

6. Conclusion

This paper presents an interpretable and personalized medical cost interval prediction framework based on multitask learning over data on mobile ends. It can predict total cost intervals based on the subcost intervals of the medical process, and the importance of each feature for cost interval prediction can be obtained using a decision tree approach based on the trained neural network's weight parameters. To begin, this paper uses a multitask learning approach to obtain the subcost intervals in the medical process and mine their correlation to exploit the value of the data; second, the subcosts pass through the full connection layer to predict the total cost intervals; finally, in order to determine the importance of patient characteristics in predicting cost intervals, the decision tree's Gini coefficient calculation method is reconstructed by using full connection layer weights of subcosts to predict total costs. Furthermore, to improve the speed of model training and convergence, the data is preprocessed using logistic regression methods, and ResNet structure is used to keep the network identity Mapping.

Data Availability

The data presented in this study are available on request from the corresponding authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Ministry of Science and Technology of Sichuan Province Program (No. 2021YFG0018, 2022YFG0038).

References

[1] S. M. Bartsch, M. C. Ferguson, J. A. McKinnell, W. O'Shea, and B. Y. SiegmundLee, "The potential health care costs and resource use associated with COVID-19 in the United States," *Health Affairs*, vol. 39, no. 6, pp. 927–935, 2020.

[2] R. Tipirneni, M. C. Politi, J. T. Kullgren, E. C. Kieffer, S. D. Goold, and A. M. Scherer, "Association between health insurance literacy and avoidance of health care services owing to cost," *JAMA Network Open*, vol. 1, no. 7, p. e184796, 2018.

[3] J. Pang, Y. Huang, Z. Xie, and Z. LiCai, "Collaborative city digital twin for the COVID-19 pandemic: a federated learning solution," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 759–771, 2021.

[4] A. Agarwal, S. Sharma, V. Kaur, and M. Kaur, "Effect of E-learning on public health and environment during COVID-19 lockdown," *Big Data Mining and Analytics*, vol. 4, no. 2, pp. 104–115, June 2021.

[5] S. Cheng, Z. Cai, J. Li, and H. Gao, "Extracting kernel dataset from big sensory data in wireless sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 4, pp. 813–827, 2017.

[6] N. Mihailovic, S. Kocic, and M. Jakovljevic, "Review of diagnosis-related group-based financing of hospital care," *Health Services Research and Managerial Epidemiology*, vol. 3, 2016.

[7] G. Robinson, M. Goldstein, and G. M. Levine, "Impact of nutritional status on DRG length of stay," *Journal of Parenteral and Enteral Nutrition*, vol. 11, no. 1, pp. 49–51, 1987.

[8] H. Fahlevi, I. Irsyadillah, M. Indriani, and S. O. Rina, "DRG-based payment system and management accounting changes in an Indonesian public hospital: exploring potential roles of big data analytics[J]," *Journal of Accounting and Organizational Change*, vol. 18, p. 4, 2021.

[9] A. A. H. de Hond, A. M. Leeuwenberg, L. Hooft et al., "Guidelines and quality criteria for artificial intelligence-based prediction models in healthcare: a scoping review[J]," *Npj Digital Medicine*, vol. 5, no. 1, pp. 1–13, 2022.

[10] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li, "Deep learning based inference of private information using embedded sensors in smart devices," *IEEE Network*, vol. 32, no. 4, pp. 8–14, 2018.

[11] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[12] L. Yang, X. Chen, Y. Luo, X. Wang, and W. Wang, "IDEA: a utility-enhanced approach to incomplete data stream anonymization," *Tsinghua Science and Technology*, vol. 27, no. 1, pp. 127–140, 2022.

[13] R. Manne and S. C. Kantheti, "Application of artificial intelligence in healthcare: chances and challenges," *Current Journal of Applied Science and Technology*, vol. 40, no. 6, pp. 78–89, 2021.

[14] V. Shankaran, S. Chennupati, H. Sanchez, L. F. Sun, and B. AlyHealeySeal, "Clinical characteristics, treatment patterns, and healthcare costs and utilization for hepatocellular carcinoma (HCC) patients treated at a large referral center in Washington state 2007-2018," *Journal of Hepatocellular Carcinoma*, vol. 8, pp. 1597–1606, 2021.

[15] B. E. Saelens, R. T. Meenan, E. M. Keast, Y. Frank, D. Kuntz, and S. P. Fortmann, "Transit use and health care costs: a cross-sectional analysis," *Journal of Transport & Health*, vol. 24, Article ID 101294, 2022.

[16] J. L. Moran, P. J. Solomon, A. R. Peisach, and J. Martin, "New models for old questions: generalized linear models for cost prediction," *Journal of Evaluation in Clinical Practice*, vol. 13, no. 3, pp. 381–389, 2007.

[17] A. I. Taloba, A. El-Aziz, M. Rasha, H. M. Alshanbari, and A. A. El-Bagoury, "Estimation and prediction of hospitalization and medical care costs using regression in machine learning[J]," *Journal of Healthcare Engineering*, vol. 2022, Article ID 7969220, 2022.

[18] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[19] I. K. Nti, J. A. Quarcoo, J. Fosu, and G. K. Fosu, "A mini-review of machine learning in big data analytics: applications, challenges, and prospects," *Big Data Mining and Analytics*, vol. 5, no. 2, pp. 81–97, 2022.

[20] M. A. Morid, K. Kawamoto, T. Ault, J. Dorius, and S. Abdelrahman, "Supervised learning methods for predicting healthcare costs: systematic literature review and empirical evaluation," in *Proceedings of the AMIA Annual Symposium Proceedings. American Medical Informatics Association*, pp. 1312–1321, Beijing China, June 2017.

[21] Z. Zheng and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[22] J. Pang, Y. Huang, Z. Xie, and Z. HanCai, "Realizing the heterogeneity: a self-organized federated learning framework for IoT," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3088–3098, 2021.

[23] J. Tie, X. Pan, and Y. Pan, "Metabolite-disease association prediction algorithm combining DeepWalk and random forest," *Tsinghua Science and Technology*, vol. 27, no. 1, pp. 58–67, Feb. 2022.

[24] B. Panay, N. Baloian, J. A. Pino, S. Peñafiel, H. Sanson, and N. Bersano, "Predicting health care costs using evidence regression[J]," *Multidisciplinary Digital Publishing Institute Proceedings*, vol. 31, no. 1, p. 74, 2019.

[25] R. Tkachenko, I. Izonin, N. Kryvinska, V. Chopyak, N. Lotoshynska, and D. Danylyuk, "Piecewise-linear approach for medical insurance costs prediction using SGTM neural-like structure," *IDDM*, vol. 21, pp. 170–179, 2018.

[26] T. Takeshima, S. Keino, R. Aoki, and K. MatsuiIwasaki, "Development of medical cost prediction model based on statistical machine learning using health insurance claims data," *Value in Health*, vol. 21, p. S97, 2018.

[27] D. Bertsimas, M. V. Bjarnadóttir, M. A. Kane, P. Kryder, and W. G. Vempala, "Algorithmic prediction of health-care costs," *Operations Research*, vol. 56, no. 6, pp. 1382–1392, 2008.

[28] S. Sushmita, S. Newman, J. Marquardt, P. Ram, V. Prasad, and A. Teredesai, "Population cost prediction on public healthcare datasets[C]," in *Proceedings of the 5th International Conference on Digital Health*, pp. 87–94, Florence, Italy, May 2015.

[29] X. Zeng, S. Moosavinasab, E. J. D. Lin, L. Simon, B. Razvan, and L. Chang, "Distributed representation of patients and its use for medical cost prediction," 2019, https://arxiv.org/abs/1909.07157.

[30] Z. He, Z. Cai, S. Cheng, and X. Wang, "Approximate aggregation for tracking quantiles and range countings in wireless sensor networks," *Theoretical Computer Science*, vol. 607, pp. 381–390, 2015.

Hindawi
Mobile Information Systems
Volume 2022, Article ID 2364154, 11 pages
https://doi.org/10.1155/2022/2364154

Research Article

Applying Deep Learning-Based Personalized Item Recommendation for Mobile Service in Retailor Industry

Minghua Xiao,[1] Qing Zhou,[1] Lei Lu,[1] Xingzhen Tao,[1] Wenting He,[1] and Youmei Zhou[2]

[1]*School of Information Engineering, Jiangxi College of Applied Technology, Ganzhou, China*
[2]*College of Architecture and Urban Planning, Tongji University, Shanghai, China*

Correspondence should be addressed to Youmei Zhou; 20310231@tongji.edu.cn

Received 12 April 2022; Revised 5 May 2022; Accepted 23 May 2022; Published 8 June 2022

Academic Editor: Yan Huang

Various kinds of mobile services allow integrating terminal customers as important coproducers into the whole retailer's business processes. People have enjoyed increasing popularity in the past years since they allow saving costs and increasing satisfaction. However, in some retail settings, as the technology relies on retailers providing terminals, it does not yet fully utilize the possibilities provided by mobile service, which until recently have mostly served as shopping aids. Recommendation systems can provide accurate recommendation services to users, especially in the field of e-commerce. In this study, a mobile retail terminal, Kkbox, leverages deep learning-based recommendation and self-service technologies to provide an express and personalized self-checkout retail environment without the engagement of storekeepers and cashiers. An attention-based mechanism for product personalization recommendation model is adopted, and it models the intrinsic relationship between users' historical interactions with products through a multilayer self-attentive network and then feeds the output of the multilayer self-attentive network into a GRU network with attention scores to model the evolution of users' interests. We analyze the performance of the product recommendation module based on user data from multiple perspectives, such as purchase frequency, purchase time, and product category. In the comparison experiments with some traditional recommendation methods, the recommendation accuracy of the model used in this study achieves better results. Besides, it significantly reduces the labor cost and provides enough flexibility. The time performance of app users is independent of store rush. The time for a transaction is significantly lower for app users than the regular shoppers during peak periods. The Kkbox has been deployed in several communities in Taizhou, China, to provide fast and convenient mobile retail services to residents.

1. Introduction

The retail sector has been playing a dominant role in economic activities, and it is a strategic industry of the national economy in China. China's total volume of retail sales increased from 9.4 trillion in 2012 to 21 trillion in 2016 [1]. The supermarket dominates the retail sector [2], and the chain operation development mode is widely adopted. In such a manner, stock can be replenished in large amounts, thus lowering the purchase cost of commodities shared by a number of chain stores in the same areas. This can also dynamically maintain the demand balance of each store [3]. The customers are able to buy most of the commodities needed in their daily life in the supermarket. Meanwhile, the large-scale and efficient management policies for supermarkets greatly reduce the price of commodities.

However, the operational efficiency of the supermarket is not as good as expected. There usually exists some frustrating experience when the customers go shopping in the supermarket, for example, the customer usually suffers from the long queue at the checkout counter, and with the investigation by researchers, such a long queue may be caused by the bar code scanning, which is time consuming [4–7]. Before completing the checkout process, all commodities need to be taken out from the shopping bag and scanned on the counter one by one. Such a process is also boring and time consuming; The hundreds of commodities always make customers get lost, whereas appropriate update news of

commodities could directly reach their potential customers and promote the purchase. But, actually, there are very limited channels (e.g., news subscription) available. Therefore, more convenient guides and assistances are expected (e.g., apps) [8–10].

Nowadays, self-service technologies (SSTs), such as self-checkout terminals, enable customers to scan and pay for groceries without interaction with the store personal [11]. SST has been proven to be an effective way to increase consumer satisfaction and convenience at checkout by avoiding the long queues associated with traditional checkout methods. [1, 2] However, SST using barcode scanning and self-service payment devices is mainly used in large shopping centers [12] and has a limited reach. In these cases, the shopping basket is transferred to a self-checkout terminal, where the actual payment and delivery of the merchandise occurs, but in practical terms, the checkout capacity remains limited during peak periods [1]. Current self-checkout solutions can be very appropriate for the digital transformation of general grocery stores, and they are already well adopted in these stores [12]. Still, they are not applicable to the convenience store environment. Unlike regular grocery stores, customers in convenience stores usually purchase very few items. In China, convenience stores are typically located in large communities with limited space and expensive rents [13, 14]. The in-store checkout load is hefty during morning and evening rush hours.

Our study combines SSTs, mobile applications, and deep learning into a solution that provides digital innovative customer experiences in the kiosk. The contributions in this work can be summarized as follows:

(i) Our solutions are driven by unique personalized content. The technology in this article is delivered via a large display in a smart retail installation and is set up to ensure that the content perfectly conforms to the user's preferences and specific interactive application.

(ii) The system collects relevant content about customer interactions and provides measurable statistics on customer engagement.

(iii) This solution can provide promoted products based on recommendations under deep learning.

The rest of the study is organised as follows: Section 2 presents the relevant work and introduces the background of our research context, questions, and methodology. Section 3 illustrates the current situation, and based on this, we design a target shopping process adapted to our research context and provide an overview of our system architecture. Section 4 discusses the results from our field study. Section 5 concludes.

2. Related Work

2.1. Self-Service Technologies. Self-service technology (SST) is a technical interface that allows customers to coproduce services without employee interaction [11]. Retailers mainly provide SST to reduce costs and improve customer experience [2]. The most commonly cited advantages of SST are convenience and speed [1]. The users' negative experience in the process of use may mainly include the experience of forced use of self-checkout, the closure of self-checkout terminals at certain times of the day (such as night), and the fact that these terminals happen to be slow when queuing [1]. Studies have shown that SST customers tend to use self-checkout terminals to check out smaller baskets and may avoid items (such as fruits and vegetables) that require additional steps in the checkout process [15, 16]. Purchasing fewer items is a crucial reason for self-checkout, as there are only long lines at the top of the shopping basket on the traditional POS machines [17]. The service quality of SST is mainly determined by function, enjoyment, design, guarantee, and convenience. SST also positively impacts customer loyalty through customer satisfaction [2]. According to the existing literature, the current technology does not provide a self-checkout solution for the grocery retail industry [18–20].

Self-checkout is a typical self-service application. Self-checkout allows retailers to order customers to scan, pack, and pay for items they want without hiring staff. Considering that retail stores alone make more than 60 billion transactions a year, 68% of which are grocery stores, gas stores, and convenience stores. This will result in significant cost savings, as self-checkout eliminates the need to hire more staff. In addition, it can also reduce the time that customers spend waiting in line, which is one of their biggest complaints. NCR (https://www.ncr.com) estimates that, on average, this technology can reduce customer queue times by 40%. From the customer's point of view, if stores minimise labor costs, self-checkout can also reduce costs for consumers. Self-checkout can provide a higher-quality consumer experience if employees are reassigned to other tasks. According to the data of an American chain grocery store, after the implementation of self-checkout, 10% of their sales came from self-checkout. They were able to transfer 7% of the front-end labor force to other stores operations [18]. Tesco has also invested heavily in self-checkout technology in its retail stores.

2.2. Development and Feasibility of New Technologies. For self-service technologies, the need for appropriate technology to improve the efficiency of the shop and optimize the consumer shopping experience has become an important issue. In this study, this problem is divided into two parts: one is the association with the goods and the other is the capture of the perceptions of people and finally their systematic design to provide solutions to the above problems. RFID is thus an important technology for correlating goods, and the deep learning-based recommendation algorithms used in online shopping are important for capturing and outputting human perceptions.

RFID and related automatic identification systems are designed to solve the problem of electronic tag technology. This type of technology allows the use of electromagnetic challenge/response exchange. This kind of technology can automatically identify objects, places, or people at a distance

without being able to remove them directly (Want, 2004). Kinsella (2003) described RFID as a simple technology. This technology enables machines to share information wirelessly. While the idea behind the technology is relatively simple, it is a system made up of tags attached to the products that send the information, card readers that receive the information the tags send, and software that collects and stores the information correctly. Both the tag and the reader are connected to an antenna that sends and receives data between the tag and the reader. Once the tag transmits its information to the reader, the reader's job is to send it to the appropriate computer device (Boyle, 2002). RFID is likely to be the next technology used on a large scale in the retail environment. RFID tags are designed to help reduce theft and better locate items. The use of labels can increase customer service, match supply to product demand, and speed delivery. Unlike bar codes, which must be passed before a scanner, RFID tags can be read remotely from a device 20 feet away. This flexibility opens up many new ways for retailers to increase CRM. RFID can also speed returns, manage warranties, and provide after-sales support. Users can also calculate costs based on RFID, especially how expensive tags are now. RFID technology can track sales after they are made.

On the other hand, in recent years, deep learning has started to be widely used in areas such as recommendation. Various deep models are able to learn discernible features from unstructured data through training. For example, Davidsan et al. proposed the "Item-KNN," which recommends items similar to items previously visited by the user, and the similarity between objects and items are expressed by the number of cooccurrences of the two [21]. Rendle et al. proposed FPMC for the following basket recommendations: FPMC uses matrix decomposition and first-order Markov chains to capture users' long-term preferences and short-term transitions, respectively [22]. BPR-MF is one of the most widely used matrix decomposition methods, which optimizes a pairwise ranking objective function by stochastic gradient descent to address the inability of traditional matrix decomposition methods to be directly applied to session-based recommendation tasks [23]. A mixed model of a shallow linear model and a deep feedforward network model has been trained, combining the memory capability of the shallow model and the generalization capability of the deep model into one, thereby balancing the accuracy and scalability of the recommendation system [24]. HRNN-Init is a personalized recommendation approach based on GRU4Rec and adds an additional GRU layer to model the evolution of user interests across sessions [25]. Zhou et al. propose a local activation unit for assigning different weights to users' historical behaviors so as to adaptively adjust the degree of influence of various historical behavioral features on the final results according to the candidate products [26, 27].

These neural networks (e.g., RNN, GRU, LSTM, and CNN) have significant advantages in modelling sequential data and have been widely used in personalized recommendations. Reference [28] pioneered the application of RNN to personalization, and their proposed GRU4Rec model showed a significant improvement over traditional methods. The CNN-based serialization recommendation embeds the items the user has interacted with into an $n^{*}k$-dimensional potential matrix and is treated as an image for processing. For example, the convolutional sequence embedding recommendation model proposed in the literature [29] consists of an embedding layer, a convolutional layer, and a fully connected layer. The convolutional layer consists of a horizontal convolutional layer and a vertical convolutional layer. In the horizontal convolution layer, all convolution results of the kF filter are maximally pooled to capture the most distinct features extracted by the filter by taking the most significant value. The operation of the vertical convolution layer is similar to that of the horizontal convolution layer. Finally, the outputs of the two convolutional layers are connected, and the tightly connected neural network layers are fed in to obtain higher-level features.

The attention mechanism affects the output by assigning different weights to the inputs. Several studies have attempted to use attention mechanisms to improve the performance and interpretability of serialized recommendations [16, 30, 31]. Reference [31] proposed a RIB model that introduces attention mechanisms into RNN-based serialized recommendation models. Reference [17] introduced an item-level attention mechanism in a local encoder model to capture the user's main intent in the current session, allowing different parts of the sequence to be dynamically and selectively input by the decoder. Besides, RNN-based recommendation models cannot model association relations at longer distances, but adding a self-attention mechanism inside the sequence can model across distances, and purely self-attention-based models (without any convolutional or recursive operations) started to be applied to the personalized recommendation. Reference [18] built a recommendation model based on a self-attention mechanism.

A review of RFID and a further review of the deep learning-based neural networks and attention mechanism reveals that both have been continually optimized in their respective fields and terms of accuracy and model, but that no research or offline consumer-oriented testing activities have been carried out to combine the technologies, while their principles and technologies show good potential for cross compatibility.

3. Design Rationale and Implementation

We designed a smart self-service kiosk that allows consumers to autonomously purchase goods in a store without a cashier or security. We leverage insights from operational patterns, and we obtain actual quantitative data from three convenient stores (e.g., 711 and FamilyMart) in China to better understand the degree and origin of the SST in a retail setting. Then, we translate what we learn in conceptual design (as illustrated in Figure 1) and implement a corresponding artefact that consists of a structure that is physically similar to a container, a WeChat-based shopping application, an in-store self-checkout accouter, and a backend operational management system. The functionalities involved are mainly grouped as follows: (1) express

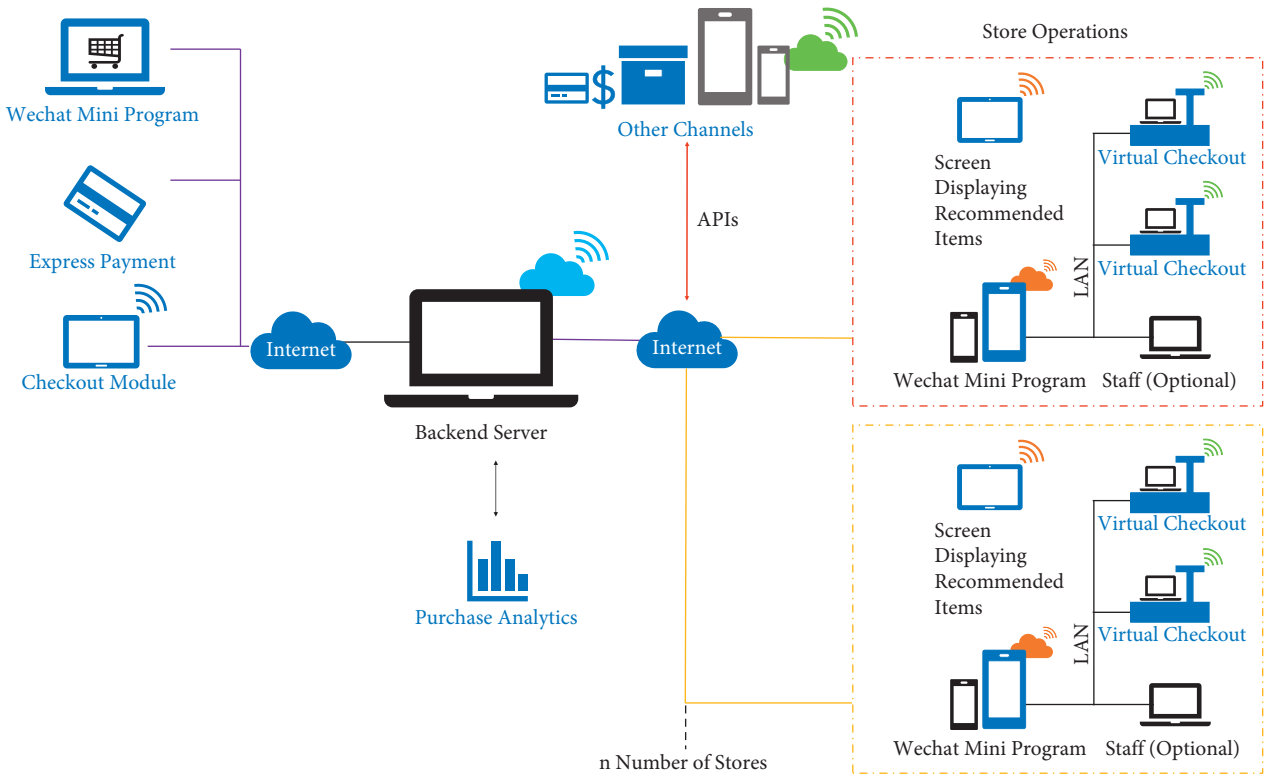


Figure 1: An overview of the system architecture of Kkbox system.

commodity checkout module, (2) user identification, and (3) operational support and shopping assistance.

3.1. Physical Design. The physical design of the kiosk requires local adjustments. Adjustments to the physical structure of the kiosk were made to accommodate the limited internal space and the SST setting. The correct placement of RFID tags is the key to successful data reading. As a result, the checkout counters and merchandise organization design will also be redesigned. The store's visual marketing identity and near transparency to the consumer should be an equal priority. Failure to follow this principle will reduce consumer comfort in an SST environment. In addition, we had to consider input/output controls for kiosks, self-checkout processes, and other parts of the organization that fit into the new setup. Based on the above-mentioned considerations, a commercially available retail kiosk is designed and manufactured, as illustrated in Figure 2.

3.2. Express Commodity Checkout Module. Kkbox's checkout module design is shown in Figure 3. RFID tags are attached to goods. Each RFID tag contains information corresponding to a unique serial number associated with the operations support system. The server's database stores the name, unit price, manufacturer, and related data of all goods.

UHD RFID could be used to read all tag information simultaneously. It can scan up to 1,000 items at one time. Equipped with an upper RFID four-wire spiral antenna, the reading distance can be extended to 10 meters. RFID readers and antennas are placed near the terminal's main entrance in the checkout area. When the goods pass through the checkout area, the reader placed in the checkout area can read the information on the RFID tag attached to the goods in a short time and transfer this information to a computer, which will retrieve the goods identified by the serial number in the goods database. It then shows the customer a list of items and the total price for further payment confirmation. The checkout page will pop up after the relevant processing process is complete.

In such a way, all the commodities can be scanned in less than 10 seconds (even if there are many commodities, for example, 100 pieces) without taking them out of the shopping bag. Besides, there will exist no queues and no access issues but 24 hrs-opening hours. Kkbox enables flexibility in stock management, inventory control, and accurate good trend analysis.

The payment system is implemented based on Alipay and WeChat Pay as an essential component, because WeChat Pay is now the most popular payment method in China, linking features such as face recognition, and it is already being used on shopping mall SST. During a customer's stay in the kiosk, his WeChat Pay account will be used as their identifier. The critical function of the fast payment module is to purchase physical products from physical retailers without interacting with the POS machine. This immediate payment process consists of four main components: (1) a user presents their WeChat Pay at the kiosk entry for authentication. (2) Users walk into the

Figure 2: A Kkbox kiosk deployed in a community in Taizhou, China.

checkout area to check the total amount and present the code for payment. (3) Our operation system initiates the payment process to payment gateways of WeChat Pay. The due amount is charged to the payment account. (4) The payment results will be sent back to the user's account and stored on the user's mobile phone or smartphone (as illustrated in Figure 3).

3.3. Shopping Assistance and Item Recommendation. Many retailers have developed "shopping assist" technology to enhance the online and in-store shopping experience. This technology can help filter and narrow down the range of products available. In addition, the technology also allows for an in-depth comparison of selected products [16]. WeChat is the most popular application, and there are over 100 million users in China. Kkbox implements a WeChat-based application (as shown in Figure 4) for shopping assistance and self-checkout, facilitated with item recommendations. It applies to the setting of administrators, arrangement of duty schedules, management of customer and member information, and control of supermarket income information. Online shopping assistance has been shown to reduce search costs and increase convenience and the quality of purchase decisions [16, 29]. Consumers increasingly utilize mobile phones in the shopping process—primarily for information search in the prepurchase phase and less for actual purchase transactions [32–35].

To achieve the item recommendation, a recommendation model (as shown in Figure 5) is adopted to achieve the personalized recommendation of goods. The model combines the multilayer self-attentive network with the AUGRU [21–24, 36]. It extracts the user's interest vector from the whole user's sequence behavior through the multilayer self-attentive network structure and AUGRU and then matches the sequence to be recommended with the user's interest through the bilinear interpolation matching function to finally obtain the sequence of goods that the user may interact with.

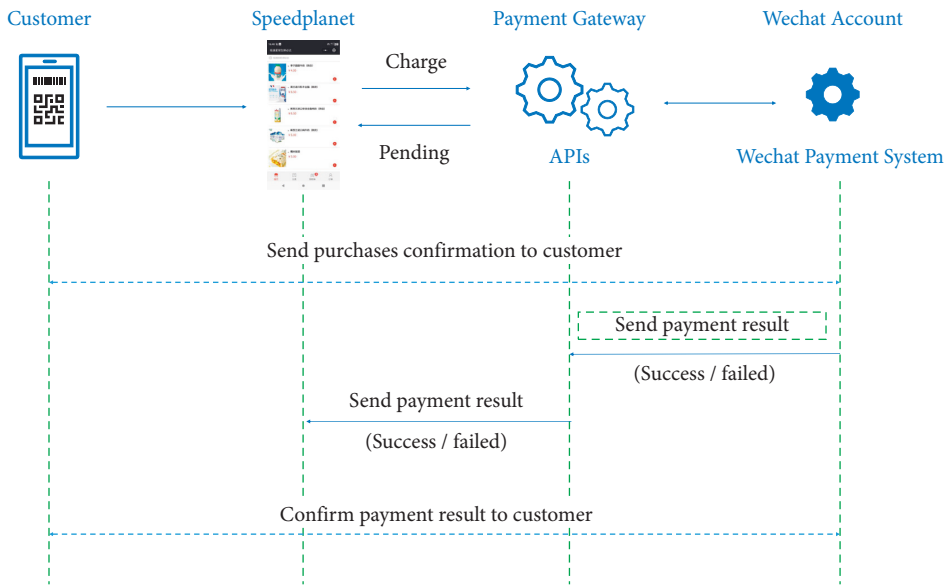


Figure 3: A fully automatic payment workflow.

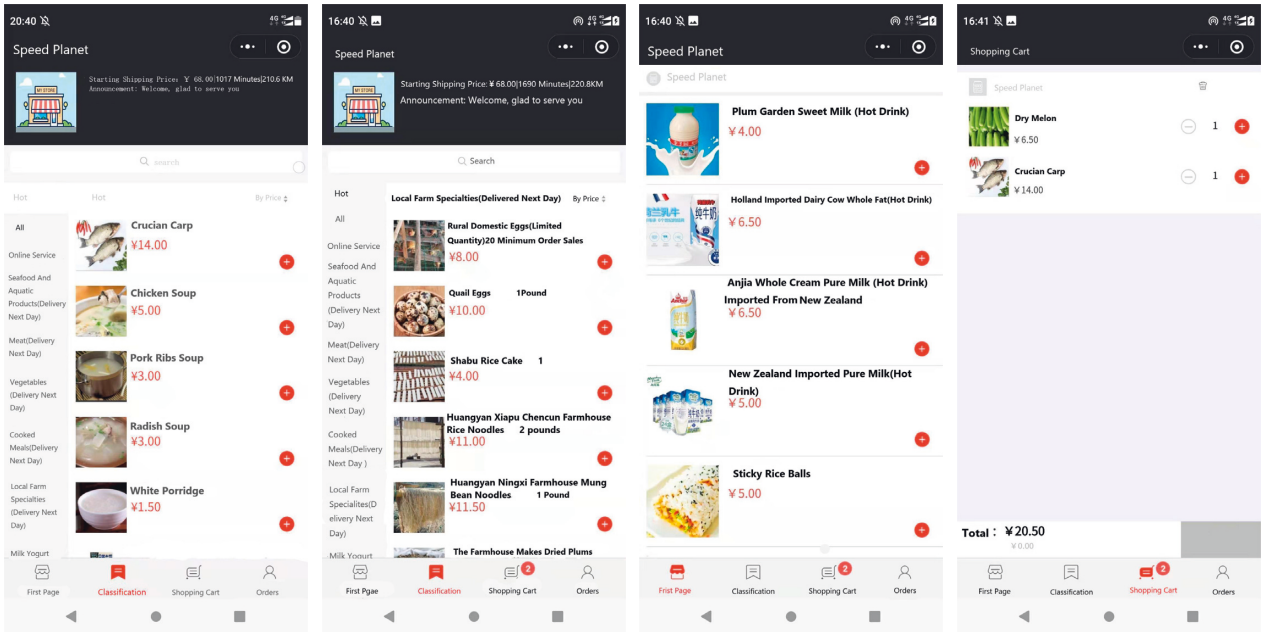


Figure 4: Screenshots of WeChat built-in app view of (1) available commodity list, (2) promotional commodity list, and (3) online shopping checkout and confirmation.

The model represents the sequence information of the goods by embedding the location codes of the historical goods, ensuring that the user's old points of interest neither disappear over time nor create unnecessary interference. The model uses a multilayer self-attentive network mechanism for the relationships between users' historical goods. The input to the encoder consists of two components: the user's historical purchase item characteristics and the purchase order of historical shopping items. By introducing location coding in the coding process, the sequence information in the user's history of clicking on goods can be effectively mined, thus enabling more effective modelling of the user's interests.

In this study, we will use a multilayer self-attention network to model the intrinsic relationship between users' historical interactions. Each layer of the self-attention network contains a query, key, and value. Their values are identical and are all vector representations of the user's historical shopping product feature vectors formed after the embedding layer. In a layer of self-attentive network, the query and key computation processes are described as follows:

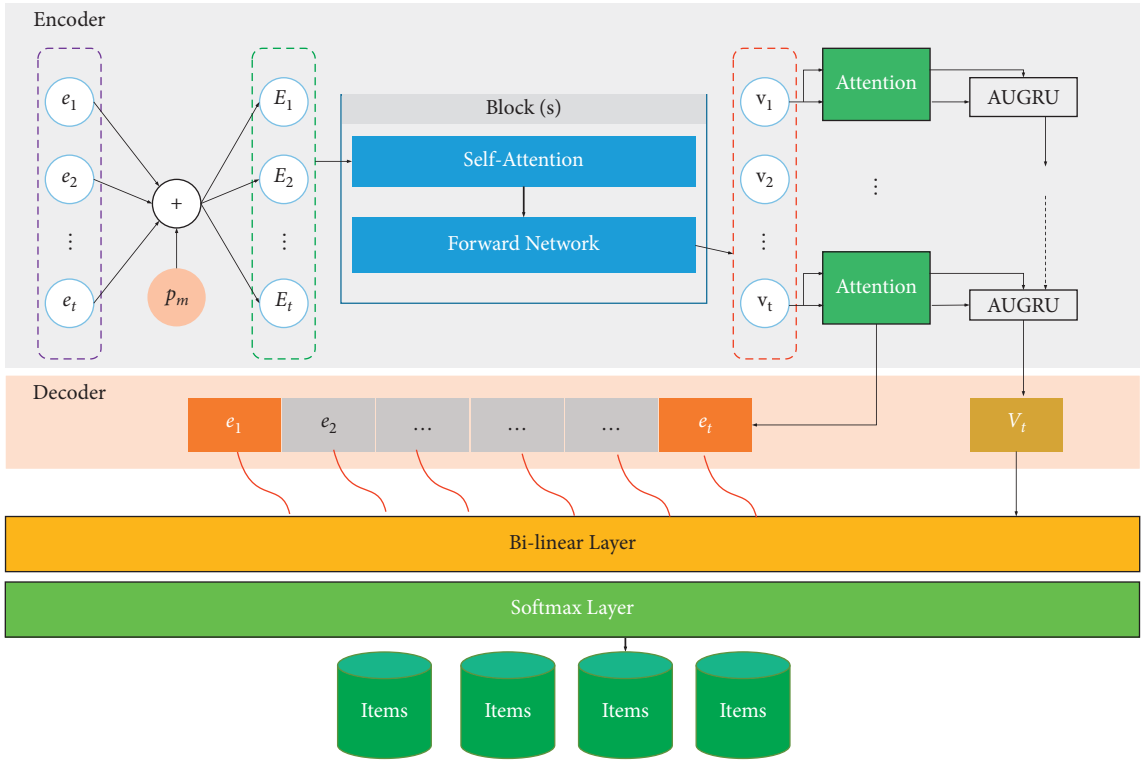


Figure 5: The insight of the item recommendation module.

$$\partial_t = \text{soft max}\left(\frac{QK^T}{\sqrt{d}}\right). \tag{1}$$

Finally, based on the obtained similarity matrix, a weighted average of each item in the historical interaction list with the vector of all historical interaction items for that user is obtained, and the representation vector of each item $\{v_1, v_2,..., v_t\}$ is as follows:

$$v_t = \sum_{i=1}^{t} \partial_i E_i. \tag{2}$$

To give the model nonlinearity and to consider the interaction between different dimensions, we input the representation vector of each clicked commodity into the feedforward network:

$$F_t = \text{ReLU}\left(v_t m + p\right)n + q, \tag{3}$$

where ReLU () is a nonlinear function, d is a scaling factor to adjust the inner product size, m and n are learnable linear matrices, and p and q are d-dimensional bias vectors. After one layer of self-attentive network, the sequence $\{v_1, v_2,..., v_t\}$ basically aggregates the embedding of the user's historical interaction goods; however, due to the diversity of goods attributes and the complexity of user interactions, it is necessary to use one or several more layers of the self-attentive network to capture more complex intrinsic relationships between interaction sequences, so we propose a multilayer self-attentive network for deeper relationship mining in the relationship extraction encoder. The kth block of self-attentive modules $(k > 1)$ can be defined as follows:

$$\begin{aligned} v_t^{(k)} &= F_t^{(k-1)}, \\ F_t^{(k)} &= \text{FFN}\left(v_t^{(k)}\right). \end{aligned} \tag{4}$$

With the multilayer self-attentive network, the model can not only capture the relationship between every two users' historical interaction goods but also focus "attention" on the goods' attributes that can really express users' interests by giving different attention scores to different goods' attributes, so that the model has the ability to capture more complex user interests. This allows the model to capture more complex user interests and facilitate the expression of user interests.

The attention function in this module is defined as follows:

$$\alpha_t = \frac{\exp\left(v_t m s_a\right)}{\sum_{j=1}^{T}\left(v_j m s_a\right)}, \tag{5}$$

where s_a denotes the embedding vector of the item to be recommended, the attention score reflects the correlation between the item to be recommended s_a and the input v_t, and a higher score indicates a stronger correlation between the two variants.

AUGRU adopts attention scores to update the state of the hidden layer of GRU to build the model of the dynamic interest of users. The bilinear interpolation matching function is used to match the user interest representation and the commodity to be recommended because it saves time and space compared to traditional methods. Given the goods that the user interacted with before moment t, the task of this model is to predict the goods that the user is about to

interact with at moment $t + 1$. In the training process, we use the ADAM to update the model parameters, and the loss function is described as follows:

$$\text{Loss}(g, u) = -\sum_{i=1}^{n} g_i \log(u_i), \tag{6}$$

where u is the true distribution of user-click sequences and g is the probability distribution of the model output results. Although the multilayer self-attentive network module can work towards the association relationship, as the number of network layers increases, the model is likely to be overfitted on the one hand. Moreover, as the number of layers increases, more parameters need to be trained, which will lead to more time and space required for the training of the model. To solve the above problem, we optimize the procedure as follows:

$$\begin{aligned} f(x) &= x + D(f(L(x))), \\ L(x) &= \gamma * * \frac{x - \lambda}{\sqrt{\mu + \varepsilon}} + \eta, \end{aligned} \tag{7}$$

where x denotes all feature embeddings for each item, $f(x)$ denotes a self-attention layer, $**$ denotes the product operation, λ represents the mean, μ denotes the variance of the feature embeddings, γ is the scaling factor, η is the bias term, and ε is a floating point number to prevent the situation that divisor is zero, respectively. That is, when using a multilayer self-attentive network, the input of each layer will be executed by a normalized layer, then the output of that layer is randomly deactivated, and the input x is added to the final output to prevent information omission.

4. Result

4.1. Experimental Setting. In this experiment, we collected the data from 6 groups of self-service kiosks for community retailing deployed in Taizhou, China. We selected a kiosk deployed in a high-density community for observation. The experimental data including total visit, customer, total amount, in-store purchase, online purchase, and shopping time are taken into consideration to evaluate the efficiency and acceptance of our proposed solution. The data collected range from July to August 2018.

Furthermore, we distribute opening flyers to passersby during the illustrated commuting hours through the part-time staff (e.g. students). We also deliver ads in WeChat moments to promote our service account on WeChat platform according to the user's location. If a customer purchase items through the app, the items would be sent to their home in one hour, but an extra delivery fee would be charged according to the purchase amount.

4.2. Conversion and Usage: Online Service Is More Attractive to Customer. Of the 500 users identified as eligible samples, 202 have purchased at least one product. There is no financial incentive for users to use the application. A total of 140 users made at least one transaction through the WeChat-based shopping app, while 26 users used the self-checkout app multiple times when purchasing products. As a result (illustrated in Figure 6), about 56% of buying users make more than one transaction. Conversion rates and utilisation rates are higher considering the recruitment channels of clients. We distinguish between proximity shopping (via flyers in stores) and WeChat online shopping. Queue types can be assigned to users based on their respective registration dates. All online customers have a higher relative conversion rate than in-store customers. Online promotions contribute more to business than regular in-store promotions.

Of all selected customers, 48 percent choose to purchase products through WeChat. Thirty-one percent prefer to use these two channels to buy products. About 38% of all face-to-face hires made purchases using the app, compared with 15% in the remote recruitment group. More impressively, 46% of face-to-face recruits cited "saving time" as their primary motivation for buying, and even 57% of those who used the app to purchase if they were in a convenience store "every day"—all (8) of those users made at least one more purchase through the app. The transactions vary in each kiosk due to their locations. However, we report an average total of 129 transactions issued in each kiosk per day. In our total sample, the most active user made a total of 31 transactions during the week, while the second most active user made a total of 26 transactions.

4.3. Time Issue. Aiming to analyzing the effects of increasing store rush and queues on the mobile app users, we analyze the time used to complete the purchase process from (1) time used to select products, (2) time for queuing if needed, and (3) checkout and payment (it is almost fixed) and its distribution during peak and nonpeak hours for all the transactions.

An in-store customer spends an average time of 107 seconds to complete the checkout and payment process (not including the time used to select commodities in the kiosk). Meanwhile, an app customer only spends an average time of 30 seconds to complete these steps but with a cost of extra charge for delivery. Comparatively, an average time of 4 minutes is required to complete the same process in a convenient store. Furthermore, we find that the mean purchase time from the app is about 390 seconds. For in-store mode, the average time used to select an item is 130 seconds. Meanwhile, the time for an app customer is over 400 seconds. We think there might be a process of making a comparison with another online e-commerce platform. Thus, the average time is longer than the in-store customer.

To compare our two metrics of the day, we relied on regular transaction data from these kiosks. We split trading into three different periods over 24 hours, with the morning peak from 1 a.m. to 7 a.m., the afternoon peak from 6 pm to 12 pm, and the rest of the day. Our results show that the mean and median of the two measures are almost equal or equal at different times of the day. We assume that the median shopping time in the afternoon and evening is slightly higher because there is less time pressure and users are more "strolling" shopping.

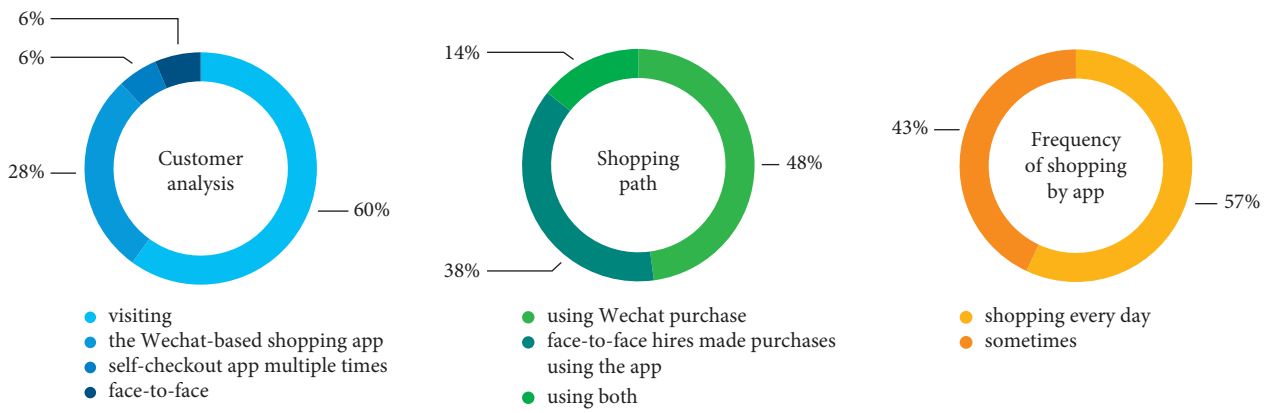


Figure 6: Analysis of customers' behaviors.

In order to avoid a broad calculation of time metrics that can be influenced by some product-specific factors, we define the calculation of the actual shopping time from the first shopping scan, which can be more accurate. We counted and calculated how long all in-store users spent shopping during the morning rush hour. The calculated average is used as a baseline and compared to the application users in the same morning session of the day. We made a total of 95 observations. The sample showed that in-store customers spent an average of 130 seconds buying, while app users spent an average of 30 seconds during the morning rush hour.

All these stores are characterized by lower unit prices and smaller items, including about 3 to 5 items per transaction and a purchase amount between 16 and 50 CNY. The share of alcohol and tobacco is 18% and 21%, respectively. The popular items include soft drinks, beer, tobacco, snacks, and bread and pastries. Our assessment also aims to understand consumer spending patterns in the current retail setting. For the pilot store, we selected that it has a central location and will be visited by more diligent customers; two of the three stores showed peak demand characteristics around working hours. The kiosk has the highest number of transactions in two periods (i.e., 7 a.m.–9 a.m. and 5 p.m.–7 p.m.). We observe that the increase during peak hours is usually more than three times the average demand. Kiosks, in particular, have long queues during peak hours, and customers face long waits. Therefore, we believe that our self-checkout module minimises the time and effort required by customers, even during peak hours, with long queues.

4.4. Item Recommendation Evaluation. We divide the data set into a training set and test set according to the time metric, in which 60% of the data is used as the training set, and the remaining 40% is further divided into 4 test sets randomly. For the evaluation of the recommendation system, Recall is an important metric, which counts the frequency of items in the test set that users actually clicked in the top K items in the recommendation list, and recall can be defined as follows:

$$\text{Recall}(K) = \frac{1}{F} \sum m_{\text{hit}}, \tag{8}$$

where K is set to 10, F is the total number of recommendations, and m_{hit} is the indicator function. If the target item appears in the current recommendation list, the value of the indicator function is 1, and vice versa is 0.

The experiments use a three-layer self-attentive network with an Adam optimization function. Table 1 lists that the model proposed in this study achieves better performance on all five datasets, indicating the effectiveness of the proposed approach. In addition, we can observe that the recommendation performance of the above models on Dataset2 is generally better than that on the other datasets, and a possible explanation for this is that Dataset3 has more users and interactions than the different datasets. At the same time, the number of items is relatively small.

As the number of self-attentive layers increases, the more thoroughly the model explores the relationships between sequences, the higher the recommendation performance. However, as the number of layers increases, the overfitting problem of the model to the training data and the excessive time and space consumed for training also occur. By observing the experimental results (in Table 2), we can learn that adding the self-attentive network to both datasets improves the evaluation indexes substantially than not adding the self-attentive network, which indicates that modelling the intrinsic relationship between user history interaction sequences is beneficial to improve on the same dataset, and the optimal performance of the model is achieved when the number of layers of the self-attentive network is 3 or 4, which means that the hierarchical self-attentive structure helps to learn more complex relationships. When the number of layers of the self-attentive network exceeds 4, the metrics of the model reach a plateau, which is due to the overfitting of the model.

Table 1: The comparison of recall results for different algorithms.

Experimental dataset	Wide and deep	DIN	FPMC	Item-KNN	BPR-MF	Proposed method
Dataset 1	0.7802	0.819	0.3901	0.2132	0.2309	0.7285
Dataset 2	0.6745	0.438	0.2904	0.4034	0.1035	0.8732
Dataset 3	0.7974	0.7252	0.1392	0.2972	0.2427	0.8029
Dataset 4	0.7189	0.5013	0.2158	0.4822	0.2521	0.8561

Table 2: The relationship between the number of the attentive network layer and algorithm performance.

Experiment group	Attentive network	Layer number	Performance
Group 1	Not engaged	0	0.1394
Group 2	Engaged	1	0.4224
Group 3	Engaged	2	0.5324
Group 4	Engaged	3	0.8013
Group 5	Engaged	4	0.8561
Group 6	Engaged	5	0.6332

5. Conclusion

We design and implement a self-service retail kiosk solution and evaluate its acceptance and practical use in a high-density community in Taizhou, China. Conclusions drawn from the usage records of 500 customers illustrate the positive value and consumer acceptance of the new retail kiosks. An entry survey provides more insight into the demographics and motivations of our study participants. It illustrates that almost half of users are often (at least once a week) unable to make purchases due to time pressures and long lines. For this, we compare it to the baseline time performance of the average in-store customer during peak hours, and the results show that the average customer saved 60 seconds by shopping at the kiosk. In addition, we can demonstrate that the purchase time required by application users was stable throughout the day, with delivery times prolonging even when queues occurred during peak morning and afternoon hours.

This study provides comprehensive guidance to better understand how to design a totally innominate self-help retail kiosk program and integrate mono in-store purchase with mobile retailing and instant delivery. This innovative practice model combines attention algorithms with RFID systems to project the technology of online product recommendations to customers into offline retail, meeting the need for more for fast consumption. Contemporary customers are basically popularised with smart terminals, which is an important basis for technological upgrades in SST, a sign of the era, and an important catalyst point in the study of smart communities that enables more ordinary residents to feel the convenience brought by the smarting of their lives.

The attempt on the algorithm is of great significance, but there are still limitations in the application research. It is suggested that future research can be carried out from the promotion of the product and the direction of multipoint multidata in line with the ability and arithmetic power of computing to optimize the research for user experience upgrading. In order to gain more understanding of application adoption and usage, as well as general consumption patterns in the current retail setting, our goal is to further expand our research with more participants over a more extended period of time and collect more data on mobile application user satisfaction.

Data Availability

The data are available from the corresponding author upon request (20310231@tongji.edu.cn).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Ministry of Housing and Urban-Rural Development (2021-K-148), General Topics of the Shanghai Philosophy and Social Science Programme (2021BCK004), and Soft Science Research Project of Shanghai 2022 Science and Technology Innovation Action Plan (22692106800).

References

[1] P. Singh, M. Gupta, A. Kumar, P. Sikdar, and N. Sinha, "E-grocery retailing mobile application: discerning determinants of repatronage intentions in an emerging economy," *International Journal of Human-Computer Interaction*, vol. 37, no. 19, pp. 1783–1798, 2021.

[2] K. Kallweit, P. Spreer, and W. Toporowski, "Why do customers use self-service information technologies in retail? The mediating effect of perceived service quality," *Journal of Retailing and Consumer Services*, vol. 21, no. 3, pp. 268–276, 2014.

[3] F. Bielen and N. Demoulin, "Waiting time influence on the satisfaction-loyalty relationship in services," *Managing Service Quality: International Journal*, vol. 17, no. 2, pp. 174–193, 2007.

[4] K. van Ittersum, B Wansink, J. M. E. Pennings, and D. Sheehan, "Smart shopping carts: how real-time feedback influences spending," *Journal of Marketing*, vol. 77, no. 6, pp. 21–36, 2013.

[5] X. Zhou, Y. Li, and W. Liang, "CNN-RNN based intelligent recommendation for online medical pre-diagnosis support," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 3, pp. 912–921, 2021.

[6] H. T. Cheng, L Koc, J. Harmsen et al., "Wide & deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pp. 7–10, Boston MA, USA, September 2016.

[7] M. Quadrana, A. Karatzoglou A, B. Hidasi B, and P. Cremonesi, "Personalizing session-based

recommendations with hierarchical recurrent neural networks," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 130–137, Como Italy, Auguest 2017.

[8] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[9] G. Zhou, X. Zhu, C. Song et al., "Deep interest network for click-through rate prediction Proceedings of the 24th," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, no. 7, pp. 1059–1068, London, UK, July 2018.

[10] X. Zhou, W. Liang, W. Li, K. Yan, S. Shimizu, and K. I. K. Wang, "Hierarchical adversarial attacks against graph neural network based IoT network intrusion detection system," *IEEE Internet of Things Journal*, p. 1, 2021.

[11] M. Shahid Iqbal, M. Ul Hassan, and U. Habibah, "Impact of self-service technology (SST) service quality on customer loyalty and behavioral intention: the mediating role of customer satisfaction," *Cogent Business & Management*, vol. 5, no. 1, p. 1, 2018.

[12] M. Hyoungeun, L. Heejung, and H. Heesup, "Self-check-in kiosk quality and airline non-contact service maximization: how to win air traveller satisfaction and loyalty in the post-pandemic world?" *Journal of Travel & Tourism Marketing*, vol. 38, no. 4, pp. 383–398, 2021.

[13] S. Gupta, S. Modgil, A. Gunasekaran, and S. Bag, "Dynamic capabilities and institutional theories for Industry 4.0 and digital supply chain," *Supply Chain Forum: International Journal*, vol. 21, no. 3, pp. 139–157, 2020.

[14] X. Zhou, W. Liang, K. I. K. Wang, and L. T. Yang, "Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations," *IEEE Transactions on Computational Social Systems*, vol. 8, no. 1, pp. 171–178, 2021.

[15] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan, "Generative adversarial networks," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–38, 2021.

[16] W. Liang, Y. Hu, X. Zhou, Y. Pan, and K. I-Kai Wang, "Variational few-shot learning for microservice-oriented intrusion detection in distributed industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5087–5095, 2022.

[17] A. Davies, L. Dolega, and D. Arribas-Bel, "Buy online collect in-store: exploring grocery click&collect using a national case study," *International Journal of Retail & Distribution Management*, vol. 47, no. 3, pp. 278–291, 2019.

[18] M. Faraoni, R. Rialti, L. Zollo, and A. C. Pellicelli, "Exploring e-Loyalty Antecedents in B2C e-Commerce," *British Food Journal*, vol. 121, no. 2, pp. 574–589, 2019.

[19] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[20] D. Li, C. Chen, Q. Lv et al., "An algorithm for efficient privacy-preserving item-based collaborative filtering," *Future Generation Computer Systems*, vol. 55, no. C, pp. 311–320, 2016.

[21] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 968–979, 2020.

[22] L. Guo, J. Chen, S. Li, Y. Li, and J. Lu, "A blockchain and IoT based lightweight framework for enabling information transparency in supply chain finance," *Digital Communication and Network*, 2022.

[23] F. Lu, Z. Zhang, and L. Guo, "A lightweight hand-crafted feature enhanced CNN for ceramic tile surface defect detection," *International Journal of Intelligent Systems*, 2022.

[24] D. Castro, R. D. Atkinson, and S. J. Ezell, "Embracing the self-service economy," *SSRN Electronic Journal*, 2010.

[25] X. Zhou, X. Xu, W. Liang, Z. Zeng, and Z. Yan, "Deep-learning-enhanced multitarget detection for end-edge-cloud surveillance in smart IoT," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12588–12596, Aug. 2021.

[26] G. Zhou, N. Mou, Y. Fan et al., "Deep interest evolution network for click-through rate prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 5941–5948, Honolulu Hawaii, USA, January 2019.

[27] Z. Cai, X. Zheng, J. Wang, and Z. He, "Private data trading towards range counting queries in internet of things," *IEEE Transactions on Mobile Computing*, p. 1, 2022.

[28] F. Liébana-Cabanillas, J. Sánchez-Fernández, and F. Muñoz-Leiva, "Antecedents of the adoption of the new mobile payment systems: the moderating effect of age," *Computers in Human Behavior*, vol. 35, pp. 464–478, 2014.

[29] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[30] B. Adipat, D. Zhang, and L. Zhou, "The effects of tree-view based presentation adaptation on mobile web browsing," *MIS Quarterly*, vol. 35, no. 1, pp. 99–121, 2011.

[31] A. A. Alalwan, "Mobile food ordering apps: an empirical study of the factors affecting customer e-satisfaction and continued intention to reuse," *International Journal of Information Management*, vol. 50, pp. 28–44, 2020.

[32] J. Davidson, B. Liebald, J. Liu et al., "The YouTube video recommendation system," in *Proceedings of the fourth ACM conference on Recommender systems*, pp. 293–296, Barcelona, Spain, September 2010.

[33] X. Zhou, X. Yang, J. Ma, and K. I. K. Wang, "Energy efficient smart routing based on link correlation mining for wireless edge computing in IoT," *IEEE Internet of Things Journal*, May 2021.

[34] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized Markova chains for next-basket recommendation," in *Proceedings of the 19th international conference on World wide web*, no. 4, pp. 811–820, Raleigh North Carolina USA, April 2010.

[35] B. Loni, R. Pagano, M. Larson, and A. Hanjalic, "Bayesian personalized ranking with multi-channel user feedback," in *Proceedings of the 10th ACM Conference on Recommender Systems*, p. 361, Boston Massachusetts, USA, September 2016.

[36] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

Hindawi
Mobile Information Systems
Volume 2022, Article ID 5056606, 7 pages
https://doi.org/10.1155/2022/5056606

Research Article

Edge Computing for Water Quality Monitoring Systems

Jianxun Ren,[1,2] Qiliang Zhu ,[1] and Changsheng Wang[3]

[1]*North China University of Water Resources and Electric Power, Zhengzhou, China*
[2]*Water Resources Information Center of Henan Province, Zhengzhou, China*
[3]*Xixiayuan Multipurpose Dam Water Diversion and Irrigation Project Construction and Management Bureau of Henan Province, Nanyang, China*

Correspondence should be addressed to Qiliang Zhu; zhuqiliang@ncwu.edu.cn

Received 22 March 2022; Accepted 6 May 2022; Published 23 May 2022

Academic Editor: Yan Huang

Aiming at the problems of insufficient real time and accuracy of water quality online monitoring and large resource consumption, this paper proposes a water quality monitoring and early warning method based on edge computing. Combined with Internet of things technology and edge computing technology, an online water quality monitoring and early warning model is designed. Through the preprocessing of the collected source data, the monitoring accuracy is improved, and edge computing technology is introduced to preliminarily analyze and process the collected data in the monitoring station, so as to save network traffic and computing resources. On the basis of online monitoring, the water quality prediction model is established by using historical water quality monitoring data to realize water quality prediction and provide a basis for staff scientific decision-making. Engineering practice shows that the model has high application value.

1. Introduction

With the rapid development of information technology, the Internet, Internet of things, cloud computing, 5G, and other technical means have been used in the collection, storage, calculation, and management of water conservancy information, which has improved the modernization level of water conservancy management ability. As a necessary means to detect the degree of water pollution and analyze the causes of water pollution, water quality monitoring is an important basis for water resources management and protection. Accurate monitoring of water resources with the help of modern emerging technologies is of great significance to improve the ecological environment [1]. In recent years, the automatic monitoring system composed of sensing technology, automatic measurement and control technology, computer technology, and communication network has achieved remarkable results [2]. However, there are still many problems in the actual production process of these technologies, which are as follows:

(i) Due to the error of sensor devices and the influence of environmental factors, there is more or less some noise in the source data collected by the water quality sensor, which is not conducive to post-processing and has an adverse impact on the accuracy of water quality monitoring.

(ii) Sensor devices based on the Internet of things generate a large amount of measurement data and transmit it to the cloud processor for analysis through the network, which will consume a lot of network traffic [3]. In addition, when the water quality is normal, the probability of data duplication is very high, so the network traffic consumed has no practical significance.

(iii) The large amount of redundant data generated by the measuring equipment brings a heavy burden to the later analysis and prediction, wastes computing resources, and has a negative impact on the real-time performance of the monitoring results.

(iv) Water quality monitoring can only reflect the current state of water quality and cannot make an effective judgment and early warning on the development trend of water quality.

The emergence of edge computing technology makes it possible to process small data generated by sensing devices in the Internet of things in real time locally [4]. This technology is a method of processing data close to the location of data generation. It adopts an open platform integrating network, computing, storage, and application core capabilities and emphasizes processing data nearby, so as to reduce the system response time, protect data privacy and security, prolong battery life, save network bandwidth, etc. [5]. At the same time, it also meets the basic needs of real-time business, application intelligence, security, and privacy protection. In the scene of water quality monitoring, although the amount of data generated by the water quality monitoring sensor equipment is small, it has high requirements for real-time processing. Therefore, the application of edge computing technology for water quality monitoring is of positive significance to improve the real-time monitoring and reduce the network flow.

In this paper, edge computing is applied in the field of water quality monitoring. Some computing tasks are unloaded to the data edge side for local processing, which can realize real time, reliable, and safe water quality monitoring and management.

2. Related Work

2.1. Water Quality Monitoring. Due to the increasingly serious water environment pollution, since the 1970s, the United States, Britain, Japan, the Netherlands, and Germany have successively established water quality monitoring systems. At present, the commonly used algorithm models mainly include the grey system model, support vector machine, multiple linear regression model, and artificial neural network [6]. Among them, the grey theory GM (1, 1) model has high requirements for the accuracy of historical data. If there is too much unknown information, the prediction error will increase and the stability of the model will deteriorate [7]. Although the multiple linear regression method has a simple principle and convenient modeling, it is mostly suitable for the application environment with the good linear condition [8]. As a very common algorithm for machine learning, SVM has the same function as a neural network and can solve many mathematical problems that cannot be solved by traditional methods. However, with the increase of data, the advantages of SVM will weaken. Due to the restriction of the Mercer condition, the selection of kernel function will be limited, which is only applicable to the modeling of small sample problems [9]. Although the neural network has the disadvantages of forgetfulness and difficulty to adjust the weight online, it has the ability to approach any nonlinear problem and can predict better on the whole.

In terms of platform design, many recent technologies use wireless sensor networks or the Internet of things as platforms for water quality monitoring and evaluation. For example, El-Deen et al. [10] proposed a low-cost wireless sensor network solution for real-time water quality monitoring, which has the advantages of low cost, lightweight, and self-organization. Tsai et al. [11] proposed a smart aquaculture system based on the Internet of things, which is used to detect the water quality of farms and provide automatic aeration to improve the survival rate of aquatic products. However, recently developed systems using wireless sensor network technology report deficiencies in energy management, data security, and communication coverage [12]. Although the Internet of things shows more efficient, safer, and cheaper advantages in application, the large amount of data generated by the Internet of things poses new challenges to data transmission and data processing.

2.2. Edge Computing. Edge computing is a research hotspot in recent years. It is a computing and network resource between the data source and cloud center, which can provide edge processing of big data [11]. In edge computing, the data will be calculated, stored, and applied at or near the Internet of things terminal. It is not necessary to upload all data to the cloud, which can effectively reduce the amount of communication transmission data. Massive data do not need centralized control decision-making. Using edge computing for distributed decision-making can reduce short-term delay and reduce user response time. Because the data are processed near the user side, it can also effectively avoid the risk of privacy disclosure caused by long-distance transmission.

Edge computing can effectively solve the problems of high latency, network instability, and low bandwidth in cloud computing. It has been applied to smart transportation, smart city, power grid detection, and other fields. In recent years, some scholars have applied edge computing technology to the field of water conservancy. Janet al. [12] combined edge computing and GIS technology to build a prediction model of ecological water demand. The model collects image data with GIS technology and divides different water resources in a timely and fast manner through edge computing data processing, so as to clarify the relationship between water resources and ecological environment. Abbas et al. [13] proposed a data link management solution based on mobile edge computing technology, which effectively realized the sinking of the service anchor and greatly shortened the service response time. Li et al. [14] proposed an incentive-based intelligent water-saving and distribution framework integrating blockchain and edge computing. The system integrates blockchain and water consumption prediction model into one framework to achieve the purpose of encouraging people to save water and prevent waste.

3. Water Quality Monitoring Model Based on Edge Computing

Aiming at the problems of insufficient real time and accuracy and high resource consumption in online water quality monitoring, this paper proposes a water quality monitoring and early warning method based on edge computing. Due to the limitation of the length of the article, we focus on the work content of the edge layer.

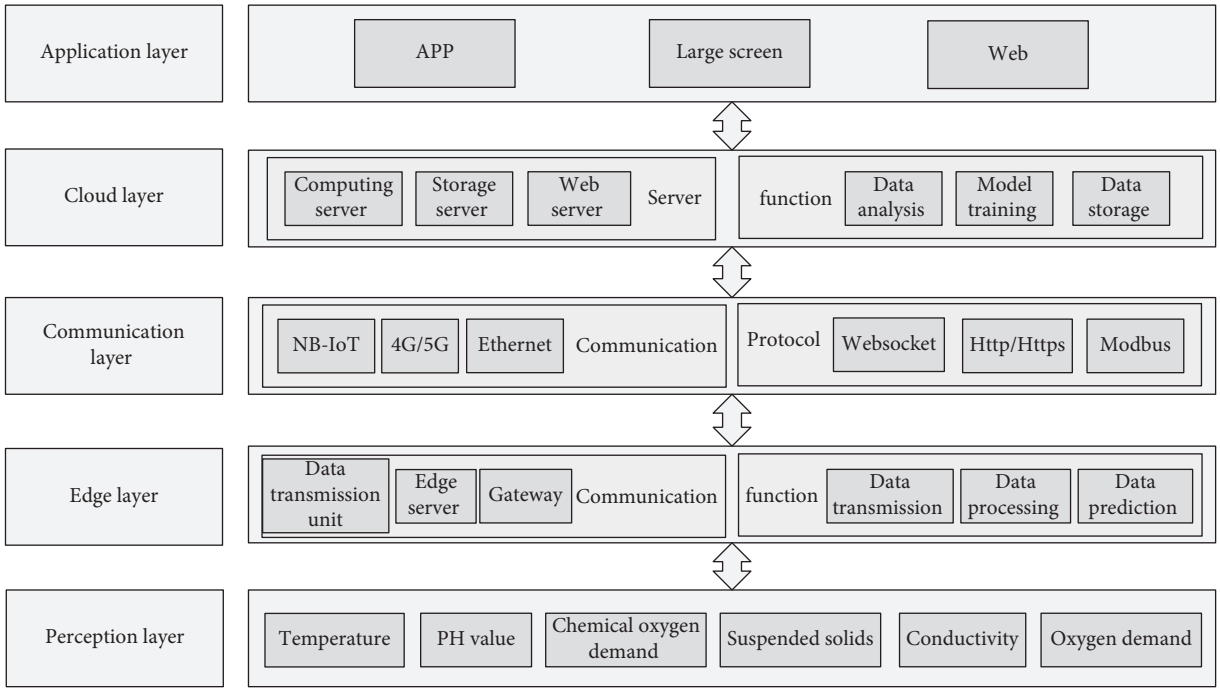


Figure 1: Water quality monitoring and early warning model based on edge computing.

3.1. Framework of the Model. As shown in Figure 1, the method model is divided into five levels, from bottom to top, including the perception layer, edge layer, communication layer, cloud computing layer, and application layer.

(i) *Perception Layer*. Data collection is the basis of informatization, and the perception layer is the cornerstone of intelligent water quality monitoring and early warning. This layer includes various Internet of things devices for water quality detection, which are used to sense water temperature, PH value, chemical oxygen demand, suspended solids, conductivity, oxygen demand, and other information to provide data sources for the system.

(ii) *Edge Layer*. The edge layer can solve the problem of data processing nearby by deploying a large number of edge nodes. On the one hand, edge nodes receive, process, and forward effective data from sensing terminals and provide certain storage, computing, and decision-making capabilities. On the other hand, the edge node interacts with the cloud platform data, uploads the optimized necessary data to the cloud, receives the calculation rules and early warning model issued by the cloud platform, and realizes the real-time prediction and early warning of water quality data. This layer mainly includes data transmission unit, edge server, gateway, and other equipment. Its main functions are data transmission, data processing, data prediction, decision-making, calculation, and unloading according to the edge computing capacity. When the water quality tends to deteriorate, it sends an early warning and uploads the monitoring results to the cloud computing layer through the communication layer according to the time upload strategy.

(iii) *Communication Layer*. Information transmission is the premise of informatization, and the communication layer is the link of information sharing. This layer is responsible for the communication between the edge layer and the computing layer. The communication modes adopted between the two layers include NB-IoT, 4G, 5G, and Ethernet and communication protocols such as WebSocket, Modbus, and http/https are mainly used for data transmission. The reliable transmission of the communication layer is an important guarantee for connecting the functions of water quality monitoring layers.

(iv) *Cloud Computing Layer*. This layer is mainly composed of a cloud computing server, data server, and web server. The cloud computing server is mainly responsible for establishing a connection with the edge layer, keeping the data transmission channel smooth, and responsible for data analysis, model training, and complex event processing. The data storage server mainly provides data storage services to store the timing data and status data transmitted by the edge layer. The web server mainly provides data query interface, security authentication interface, and device control interface to provide services for the application layer.

(v) *Application Layer*. The application layer analyzes and arranges according to the interface provided by the webserver. According to different user roles, it displays to users through mobile app, web service,

and smart screen, displays water quality monitoring data in real time, and sends out an early warning in time. This aims to realize user operation business, which can improve the level of scientific decision-making and intelligent management.

3.2. Preprocess the Original Data. The function of data preprocessing is to eliminate the interference caused by equipment vibration, environmental change, and other factors in the data collection process of water quality sensors and reduce errors, so as to improve the accuracy of water quality monitoring and the accuracy of the prediction model. Data preprocessing mainly includes the removal of abnormal values of original data and the correction of abnormal values and provides basic data for subsequent water quality monitoring data analysis and prediction model establishment. In this paper, the quartile method is used to identify outliers.

The quartile method is to arrange all the data into an ascending sequence $X = [x_1, x_2, \ldots x_{n-1}, x_n]$, where $x_i < x_{i+1}$, n is the total number of samples, and x_i represents a point in the sequence. Sequence X is divided into four equal parts, in which each part contains 25% data. The dividing points of each data are the lower quartile q_1, median q_2, and upper quartile q_3, respectively. IQR is called interquartile interval, which is the difference between the upper quartile q_3 and the lower quartile q_1. The data in IQR account for half of all the data in the sequence. q_1, q_2, q_3 are calculated as follows:

$$q_2 = \begin{cases} x_{n+1/2} & n = 2k+1; k = 0, 1, 2, \ldots \\ \dfrac{x_{n/2} + x_{n+2/2}}{2} & n = 2k; k = 0, 1, 2, \ldots \end{cases}. \quad (1)$$

When $n = 2k\,(k = 1, 2 \ldots)$, divide X into two parts from q_2, where q_2 is not included in the two parts, and calculate the median q_2^a and q_2^b of the two parts, respectively, and then, $q_1 = q_2^a$, $q_3 = q_2^b$.

When $n = 4k + 3\,(k = 1, 2 \ldots)$, then

$$\begin{cases} q_1 = \dfrac{3}{4}x_{k+1} + \dfrac{1}{4}x_{k+2}, \\ q_3 = \dfrac{1}{4}x_{3k+2} + \dfrac{3}{4}x_{3k+3}. \end{cases} \quad (2)$$

When $n = 4k + 1\,(k = 1, 2 \ldots)$, then

$$\begin{cases} q_1 = \dfrac{1}{4}x_k + \dfrac{3}{4}x_{k+1}, \\ q_3 = \dfrac{3}{4}x_{3k+1} + \dfrac{1}{4}x_{3k+2}. \end{cases} \quad (3)$$

Finally, the interquartile distance can be calculated as follows:

$$IQR = q_3 - q_1. \quad (4)$$

The limits of outliers in the data sample are as follows:

$$[F_1, F_u] = [q_1 - 1.5IQR, q_3 + 1.5IQR]. \quad (5)$$

If $x \notin [F_1, F_u]$, determine $x_i \triangleright$ as abnormal data; on the contrary, it is determined that x_i is normal.

In this paper, the polynomial fitting method is used for the secondary identification and elimination of outliers. After the two eliminations of outliers, the outliers can be regarded as missing values, so the correction of outliers is to fill in the missing values. Therefore, the polynomial fitting of the sequence after eliminating outliers twice and filling the missing values with the fitted corresponding values can achieve the purpose of correction. The goal of the polynomial fitting is to minimize the sum of squares of errors. Suppose a function combination

$$f(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + \cdots + a_m\varphi_m(x). \quad (6)$$

The sum of squares of errors is expressed as

$$\|\delta\|^2 = \sum_{i=1}^{n} \delta^2 = \sum_{i=1}^{n} [f(x_i) - y_i]^2, \quad (7)$$

where $(x_i, y_i)\,(i = 0, 1, \ldots n)$ is a given set of data. Polynomial fitting is to find a curve that is closest to all data points on the premise of minimizing the sum of squares of errors, that is, to find $f(x)$ that minimizes $\|\delta\|^2$.

The process of finding the fitting curve function is transformed into the problem of finding the minimum value of the multivariate function $H(a_0, a_1, \ldots a_m)$, and the multivariate function is expressed as follows:

$$H(a_0, a_1, \ldots a_m) = \sum_{i=1}^{n} \delta^2 = \sum_{i=1}^{n} [f(x_i) - y_i]^2. \quad (8)$$

By solving the minimum value of the multivariate function, the solution $(a_0^*, a_1^*, \ldots a_m^*)$ can be obtained, so that the least square solution of the function $f(x)$ is $f(x) = a_0^*\varphi_0(x) + a_1^*\varphi_1(x) + \cdots + a_m^*\varphi_m(x)$. Generally, the fitting degree of the polynomial fitting is 3 times. If it is less than 3 times, the peak of the curve may be lost; if it is higher than 3 times, the fitting time is too long, and it is easy to produce false peaks. In this paper, the fitting degree of the polynomial fitting is set to 3.

3.3. Task Migration and Data Transmission. Different from the traditional method of transmitting the collected data to the cloud for analysis, this topic uses the edge computing technology to analyze and process the collected data at the near end. After preprocessing the data collected by the perception layer, data calculation, state recognition, result transmission, and data prediction are carried out in the edge layer. When the processing capacity of the edge layer is insufficient, the computing task will be unloaded to the cloud. Figure 2 illustrates the process of task unloading and data transmission.

(i) *Task Migration.* The edge layer processes the data according to its own processing capacity. If its processing capacity is sufficient, the data processing will be carried out in the edge layer. On the contrary,

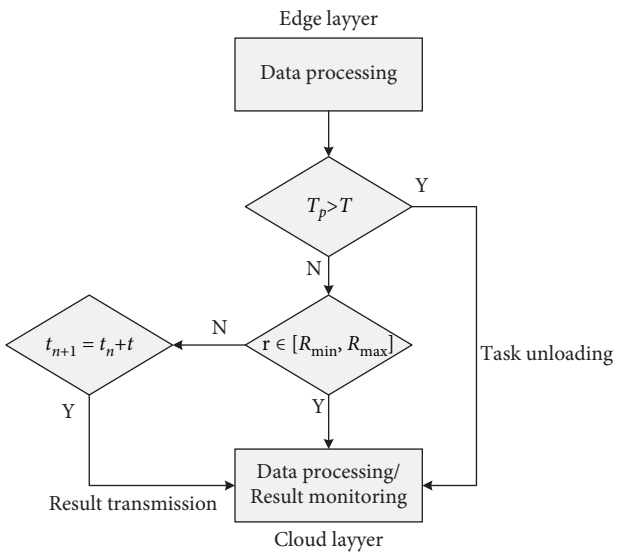


Figure 2: Task migration and data transfer process.

it will send the collected data to the cloud for processing. In the process of operation, we judge whether task migration is necessary according to the time of data processing. Suppose that the processing time of the task is T_p and the threshold of the processing time is T. If $(T_p) > T$, it means that the task processing times out, and the computing task will be unloaded to the cloud for execution.

(ii) *Abnormal Submission*. If the calculation task is within the capability of the edge layer, the water quality detection results will be completed in the edge layer. Assuming that R is a water quality detection result, $[R_{\min}, R_{\max}]$ represents the normal range of the result. If $r \notin [R_{\min}, R_{\max}]$, it indicates that the water quality is abnormal, and the water quality detection data will be transmitted to the cloud for further calculation and confirmation. If it is indeed an abnormal water quality, the early warning system is started.

(iii) *Result Upload*. If $r \in [R_{\min}, R_{\max}]$, the system will judge whether a time period has been reached. Suppose t represents a result transmission time period, and t_n represents the time of uploading the result for the nth time. If $t_{n+1} = t_n + t$, then the water quality detection results are uploaded to the cloud. Otherwise, the results will not be uploaded, but the next round of detection results will be calculated until a new upload cycle is reached. This way of periodically uploading results can not only ensure the consistency of detection data but also greatly save bandwidth and reduce data redundancy.

In short, once the processing capacity of the edge layer is insufficient, the computing task will be unloaded to the cloud for execution. When the water quality is normal, it is transmitted to the remote end according to the set frequency. When the water quality data are abnormal, it will be transmitted to the cloud in real time. This mode will greatly save bandwidth and ensure the real-time upload of detection results.

3.4. Water Quality Prediction. Water quality prediction is an important module of the model. Due to the variety and complexity of water environmental factors and the complex nonlinear relationship, many experts have focused on nonmechanistic water quality models and made some progress. However, many of these models have the disadvantage of large error. Considering that the artificial neural network model has strong adaptability, self-learning, and high fault tolerance, this paper uses BP neural network to predict water quality. BP neural network algorithm is a multilayer feedforward network trained by error backpropagation algorithm. As shown in Figure 3, the topology of a multilayer BP neural network consists of three layers: input layer, output layer, and hidden layer, in which there can be more than or equal to one hidden layer. There are several neurons in each layer. The neurons in the same layer have no relationship with each other, but only have input-output relationship with the neurons in adjacent layers. In order to improve the global convergence of the BP neural network, we use the hybrid optimization method based on the Nelder–Mead simplex method and cuckoo search algorithm to optimize the weight and deviation of the BP network. For details, refer to our previous work [15].

3.4.1. Determination of Input and Output Layer. The indicators used for water quality evaluation are water temperature, PH value, chemical oxygen demand, suspended solids, conductivity, oxygen demand, etc. In the water quality prediction model, the data of six indicators are normalized as the model input nodes and the water quality grade as the model output node. Therefore, the number of input layer nodes of the BP neural network is 6, and the number of output layer nodes is 1.

3.4.2. Hidden Layer Determination. According to Kolmogorov's theorem and the least square approximation theorem of mapping, when the number of neurons is enough, the BP neural network with hidden layers of 1 can approximate any nonlinear function [16]. Therefore, the hidden layers of the BP neural network prediction model are determined as 1.

Determining the number of hidden layer nodes is a very important step in the process of initializing the network structure. Too many nodes in the hidden layer will increase the amount of calculation of the BP neural network and easily lead to the overfitting problem; if the number of hidden layer nodes is too small, it will affect the performance of network training and fail to achieve the expected effect. In order to improve the speed of network learning, this paper determines the best number of hidden layer nodes by comparing the error of verification set under different number of nodes.

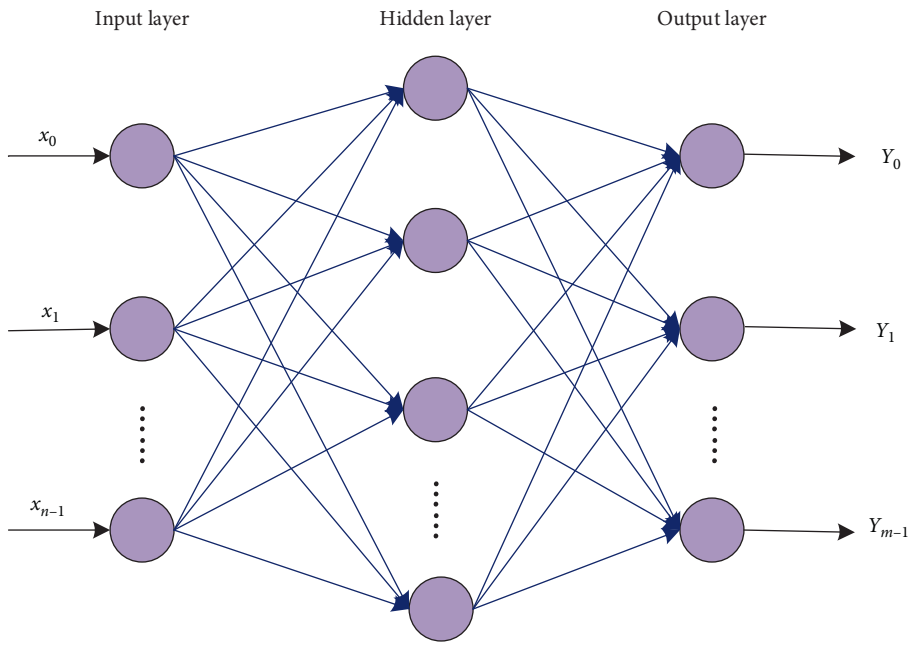


Figure 3: Topology of three-layer BP neural network.

3.4.3. Selection of Excitation Function. Sigmoid function is one of the most commonly used activation functions in the BP neural network, which is closest to biological neurons in the physical sense. It can compress a real value to the range of [0, 1] and can keep the data amplitude from large changes. The function of the sigmoid function can meet the application requirements of the solution in this paper. Therefore, the sigmoid function is selected as the activation function in this paper. The detailed description is as follows:

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{9}$$

When x approaches negative infinity, $f(x)$ approaches 0; when x approaches positive infinity, $f(x)$ approaches 1; when $x = 0$, $y = 1/2$.

3.4.4. Weight and Deviation Initialization. The weight threshold initialization of the BP neural network is usually realized by generating random numbers. In this paper, the weights and deviation random numbers between (0, 1) are randomly generated, and then, the hybrid optimization algorithm based on the Nelder–Mead simplex method and the cuckoo search algorithm is used to obtain the final initialization weights and deviation values.

An improved neural network algorithm is proposed to establish a water quality prediction model. Using the preprocessed historical data to train the prediction model, get the optimal parameters of the model, verify the trained model, and evaluate and analyze the prediction results.

4. Engineering Application

This paper takes the water conveyance and irrigation project of the Xixiayuan water control project in Henan Province as the research object. The function of the Xixiayuan project is mainly reverse regulation, combined with power generation and comprehensive utilization of water supply and irrigation. While ensuring the continuous flow of the Yellow River, it fundamentally eliminates the adverse impact of peak shaving of Xiaolangdi Hydropower Station on the downstream river and plays a vital role in ecological, environmental protection, and industrial and agricultural production water.

Table 1: Data transmission response time.

Deployment scheme	Average response time (ms)
Local computing	33
Cloud-edge computing	36
Cloud computing	113

Experimenters deployed water quality detection sensors and edge gateways in the canal. The sensor is connected to the edge gateway through RS485 serial port line. The edge gateway can upload data to the cloud through mobile communication, and the content of the cloud platform can be viewed in real time through the web. The cloud platform sets water quality early warning rules and triggers early warning information according to the data uploaded by edge nodes. Table 1 shows the average data transmission response time of different deployment schemes. We can see that the average response time of the cloud side system computing scheme proposed in this paper is 36 ms, which is basically consistent with that of the local computing scheme, but far lower than that of the cloud computing mode. It shows that the model proposed in this paper makes a significant contribution to the real-time performance of data transmission. In terms of water quality prediction, the accuracy of using the BP neural network based on the original collected data is 88.92%, which is much higher than that of the multiple linear regression prediction method. The accuracy

of using the BP neural network based on the preprocessed data is 91.16%. Obviously, the prediction model and data preprocessing method proposed in this paper have also achieved ideal results.

5. Conclusion

In order to improve the real-time transmission of water quality monitoring data and the accuracy of water quality early warning, a water quality monitoring and early warning model based on edge computing is proposed in this paper. The model makes full use of the computing power of the edge layer to process the water quality detection data collected by the perception layer. According to the actual needs, this paper designs the task migration and data transmission rules of the edge layer and puts forward the methods of data preprocessing and water quality prediction. Engineering practice shows that the model and method proposed in this paper have high application value. In future research, we will pay more attention to the research of water quality early warning and the application of edge computing in safety supervision [17, 18].

Data Availability

The data used to support the findings of this study have been deposited in the Baiduyun (https://pan.baidu.com/s/1I7-G1NR3TVXC9gWjmmMaFQ?pwd=mo00).

Conflicts of Interest

The authors declare do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

Acknowledgments

This work was supported by the high-level talents research initiation project of North China University of Water Resources and Electric Power (No. 201811026) and the Science and Technology Project of Henan Province (No. 222102240010).

References

[1] J. O. Ighalo, A. G. Adeniyi, and G. Marques, "Internet of things for water quality monitoring and assessment: a comprehensive review," *Artificial Intelligence for Sustainable Development: Theory, Practice and Future Applications*, vol. 912, pp. 245–259, 2021.

[2] Y.-S. Lin, Y. F. Lin, A. Nain, Y. F. Huang, and H. T. Chang, "A critical review of copper nanoclusters for monitoring of water quality," *Sensors and Actuators Reports*, vol. 3, Article ID 100026, 2021.

[3] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *Proceedings of the IEEE 39th International Conference on Distributed Computing Systems*, pp. 144–153, (ICDCS), Dallas, TX, USA, July 2019.

[4] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," in *Proceedings of the IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 968–979, Manhattan, NY, USA, May 2020.

[5] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: a survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[6] R. P. N. Budiarti, A. Tjahjono, M. Hariadi, H. P. Mauridhi, and Development of IoT for Automated Water Quality Monitoring System, in *Proceedings of the International Conference on Computer Science, Information Technology, and Electrical Engineering*, pp. 211–216, (ICOMITEE), Jember, Indonesia, October 2019.

[7] W. Zhai, X. Zhou, J. Man et al., "Prediction of water quality based on artificial neural network with grey theory," *IOP Conference Series: Earth and Environmental Science*, vol. 295, no. 4, Article ID 042009, 2019.

[8] J. Zhu, P. Sun, Y. Gao, and P. Zheng, "Clock differences prediction algorithm based on EMD-SVM," *Chinese Journal of Electronics*, vol. 27, no. 1, pp. 128–132, 2018.

[9] W. C. Leong, A. Bahadori, J. Zhang, and Z. Ahmad, "Prediction of water quality index (WQI) using support vector machine (SVM) and least square-support vector machine (LS-SVM)," *International Journal of River Basin Management*, vol. 19, no. 2, pp. 149–156, 2021.

[10] S. K. N. El-Deen, H. Elborai, H. E. M. Sayour, and A. Yahia, "Wireless sensor network based solution for water quality real-time," *Monitoring" Egyptian Journal of Solids*, vol. 41, pp. 49–62, 2018.

[11] K. L. Tsai, L. W. Chen, L. J. Yang, H. J. Shiu, and H. W. Chen, "IoT based Smart Aquaculture System with Automatic Aerating and Water Quality Monitoring," *Journal of Internet Technology*, vol. 23, no. 1, pp. 177–184, 2022.

[12] F. Jan, N. Min-Allah, and D. Düştegör, "IoT based smart water quality monitoring: recent techniques, trends and challenges for domestic applications," *Water*, vol. 13, no. 13, p. 1729, 2021.

[13] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," in *Proceedings of the IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, Manhattan, NY, USA, September 2017.

[14] Y. Li, J. Xie, R. Jiang, and D. Yan, "Application of edge computing and GIS in ecological water requirement prediction and optimal allocation of water resources in irrigation area," *PLoS One*, vol. 16, no. 7, Article ID e0254547, 2021.

[15] D. H. Fan and S. Gao, "The application of mobile edge computing in agricultural water monitoring system," *IOP Conference Series: Earth and Environmental Science*, vol. 191, no. 1, Article ID 012015, 2018.

[16] T. Thakur, A. Mehra, V. Hassija et al., "Smart water conservation through a machine learning and blockchain-enabled decentralized edge computing network," *Applied Soft Computing*, vol. 106, Article ID 107274, 2021.

[17] Q. Zhu, S. Wang, Q. Sun, H. H. Ching, and Y. Fangchun, "Service classification based on improved BP neural network," *Journal of Internet Technology*, vol. 19, no. 2, pp. 369–379, 2018.

[18] H. Quan, D. Srinivasan, and A. Khosravi, "Short-term load and wind power forecasting using neural network-based prediction Intervals," in *Proceedings of the IEEE Transactions on Neural Networks & Learning Systems*, vol. 25, no. 2, pp. 303–315, Manhattan, NY, USA, August 2013.

Hindawi
Mobile Information Systems
Volume 2022, Article ID 9103437, 14 pages
https://doi.org/10.1155/2022/9103437

Research Article

A ResNet-LSTM Based Credit Scoring Approach for Imbalanced Data

Anqin Zhang,[1] Baicheng Peng,[1] Jingjing Chen,[2] Qingfu Liu,[2] Shibo Jiang,[3] and Youmei Zhou[4]

[1]*College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai, China*
[2]*School of Economics, Fudan University, Shanghai, China*
[3]*School of Hotel and Tourism Management, The Hong Kong Polytechnic University, Hong Kong SAR, China*
[4]*College of Architecture and Urban Planning, Tongji University, Shanghai, China*

Correspondence should be addressed to Youmei Zhou; 20310231@tongji.edu.cn

Received 23 February 2022; Revised 21 March 2022; Accepted 29 March 2022; Published 26 April 2022

Academic Editor: Yan Huang

Detecting potential defaults or bad debt with limited information has become a huge challenge. The main difficulties faced by the credit scoring are sample imbalance and poor classification performance. For this reason, we first proposed the auxiliary conditional tabular generative adversarial network (ACTGAN) to generate sufficient default transaction samples from the original data, then we designed a model based on ResNet-LSTM used for feature extraction, which includes two submodels of ResNet and LSTM to extract static local features and dynamic temporal features from the original data, respectively. After that, a spatiotemporal attention module is added to calculate the importance of the two submodel's output in order to extract more critical information. Finally, we applied the focus loss function into the XGBoost classifier to improve the probability output of the credit default risk. We verified the designed credit scoring model in two real-world datasets. The experimental results showed that ACTGAN can effectively solve the problem of data imbalance. The ResNet-LSTM+XGBoost model for classification is better than other traditional algorithms in F1 value, AUC, and KS value, which proves the effectiveness and portability of this model in the field of credit scoring.

1. Introduction

With the advancement of computer science and technology, online financial service is booming worldwide. While the industry is booming, the amount of data in the financial industry is showing explosive growth, and the increase in consumer credit demand has also brought huge amount of financial fraud incidents, which will seriously damage consumers and financial platforms. Therefore, in order to minimize the losses of platforms and consumers, many researchers have conducted studies, and proposed a large number of models to predict the credit risk level of online loan customers and avoid the occurrence of default or bad debt.

Credit scoring can be summarized as a binary classification problem, which predicts the default probability of loan applicants and divides a loan into default or nondefault [1], thereby helping financial institutions make appropriate decisions. Meantime, credit scoring is also an imbalanced classification problem. The number of default samples is significantly lower than the normal nondefault samples. However, the identification of default samples is what we should focus on [2]. The misclassification of fraudulent transactions as normal is much more damaging than detecting a normal transaction as fraud in credit scenario [3]. Therefore, researchers are trying to improve the accuracy rate of default sample classification in recent years. The current mainstream methods for dealing with imbalanced datasets can be divided into three ways [4]: the first is based on data preprocessing, such as undersampling and oversampling methods, by eliminating samples from the majority class or increasing samples from the minority class to change

the proportion of the original imbalanced data. Although these methods alleviate the problem of data imbalance to some extent, deleting the majority class samples inevitably causes information loss and makes the model unable to use the existing information. On the other hand, the method of adding samples to the minority class lacks data diversity, cause overfitting of the model to a certain extent.

The second way pays attention to feature extraction utilizes machine learning techniques to mine hidden features of data. Models such as multilayer perceptron [5], convolutional neural network [6], and deep belief network [7] have been widely used in data mining in the field of credit scoring. However, an obvious drawback is that existing models exclusively regard static features and dynamic features as a whole feature space as the input of neural network, and ignored time dependencies of user behavior data. For this reason, our proposed model incorporates both static features and dynamic time-based data into model input, which captures critical information from multi-source heterogenous credit data.

The third way aims to improve the performance of classification algorithm, among which cost-sensitive learning and ensemble learning methods are particularly prominent. The cost-sensitive learning reduces the biased error towards the negative class and improves the recall rate of the positive class. Ensemble learning is a machine learning approach where multiple learners are trained to solve the classification problem. The central concept is to combine several "weak leaners" into a "strong learner", thereby eventually boosting the performance of classifiers. However, the improvement at the algorithm level only assigns more classification weights to the minority class samples, which prone to overfitting, and does not solve the problem of the scarcity of positive samples.

Since it is difficult for a single method to satisfy the requirements of different imbalanced datasets, the applicability is generally not strong. At the same time, the combined model can take advantages of each single credit scoring method, thereby our research considers all three levels of imbalanced credit scoring. We firstly propose an auxiliary conditional tabular generative adversarial network (ACTG AN) to alleviate the class imbalance problem in credit scoring task. Specifically, our ACTGAN which uses the Wasserstain distance to define the difference between the real data and the generated data. An auxiliary classifier is added to discriminator to stabilize generator's output. Gradient penalty is introduced to optimize the loss function. Crosslayer is added to the network to calculate high-dimensional feature interactions and generating a sufficient number of positive samples to form an enhanced dataset. Results show that ACTGAN can significantly improve the classification performance of the credit default prediction model.

Secondly, a hybrid deep learning feature extraction algorithm is designed, which divided data features into static local features and dynamic temporal features. Static financial data is input into a convolutional neural network with residual module, and dynamic feature data is input to LSTM to capture key information on its time series, and attention module is used to calculate the importance of static feature vectors and dynamic feature sequences. The fully connected layer is then applied to fuse two kinds of feature embeddings into a unified latent feature space. Finally, at the algorithm level, the Focal Loss for the imbalanced data classification is used to improve XGBoost and obtain the final output.

The main contributions of our research are as follows. First, a new generative adversarial network ACTGAN is proposed to generate more minority class samples to overcome the class imbalance problem. Second, we employ a feature extraction module to integrate multi-source heterogeneous data into the latent feature space and use attention layer to calculate the importance of information. Third, at the classification algorithm level, we leverage a focal loss function to improve the XGBoost classifier.

The remainder of the paper is structured as follows: Section 2 details related previous literatures on credit scoring. Section 3 describes the proposed framework. Section 4 presents the statistical information of datasets, the evaluation experiments and the result summarizations. The last section draws conclusions and future research directions.

2. Theory and Methods

2.1. GAN for Data Generation. Generative adversarial network (GAN) is an excellent generative model network proposed by Goodfellow et al. [8]. A typical GAN consists of two components: a generator G learns the probability distribution of real data and transforms the input noise z into new synthetic data, a discriminator D tries to distinguish the real samples from generated ones. The competition between G and D can be formalized as a minimax game:

$$\begin{aligned} Score &= \min_{G} \max_{D} V(D, G) \\ &= E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log (1 - D(G(z)))] \end{aligned} \tag{1}$$

In formula (1), p_{data} represents the distribution of real data. p_z represents the distribution of input random noise, $G(z)$ represents generated data, $V(D, G)$ represents the output value of D and G. We train the D to maximize $V(D, G)$. We train the G to minimize $V(D, G)$. Current studies [9] [10] adopt GAN as the over-sampling approach for the minority class to solve the class imbalance problem caused by credit data. They first train the GAN on a specific dataset and then use the well-trained Generator G to produce new synthetic samples. Finally, the original real samples are mixed with the generated ones and used to train the classifier for credit scoring. Although these methods have made remarkable progress, GANs with the vanilla GAN loss are difficult to train. Since the input of the GAN samples from random noise, the generated data is also random and chaotic, and it is impossible to control what category the generated image or data belongs to, which causes model collapsing and nonconvergence. Hence, We extend GAN to solve the class imbalance and mode collapsing in credit

scoring by adding conditional restrictions and gradient penalty.

2.2. Deep Learning for Feature Extraction. Inspired by the success of deep learning in a series of fields, several models started to employ neural networks for credit scoring. Based on the multilayered perceptron (MLP) approach, Blanco et al. [11] built several nonparametric credit scoring models and showed demonstrated excellent performance against traditional models. Metawa et al. [12] utilize a deep belief network (DBN) to model rich and complicated information for credit scoring. Deng et al. [13] adopted convolutional neural network (CNN) to capture the relations among the chosen attributes and output the default probabilities. With the application of deep learning technology, many cross domain research methods are gradually introduced into the field of credit evaluation, such as natural language processing to mine the correlation between lending companies [14], graph neural network to mine the relationship between entities in the social network of credit users [15–17], the autoencoder [18, 19] uses encoding-decoding technology to achieve data dimensionality reduction and other operations, which can achieve deeper mining of data patterns.

2.3. Imbalance Classification. The current imbalanced data processing methods based on ensemble learning mainly combine ensemble learning with other imbalanced data classification processing methods to comprehensively improve the classification effect. For example, Iranmehr et al. [20] proposed a cost-sensitive learning-based structured SVM ensemble classification algorithm, which increased the weight of minority samples and improved the classification accuracy of unbalanced data. Paleologo et al. [21] extended the advantage of the bagging approach, where the training subsets are formed by random sampling to address a class of imbalanced problems. Luo [22] compared the performance of the bagging approach using DT, SVM, K-nearest Neighbor (KNN), and MLP based on an imbalanced and large dataset. They obtained that bagging KNN was more sui than other methods for large and imbalanced datasets in credit scoring. Wang et al. [23] proposed a classification algorithm that combines the undersampling method and cost sensitivity, which improves the classification performance on unbalanced data. Tsai et al. [24] conducted a comprehensive study comparing classifiers ensemble methods for three public credit scoring datasets.

3. Methodology

3.1. ACTGAN. In order to solve the problem of data imbalance, based on the conditional generative adversarial network [25], we proposed a new generative adversarial network structure (ACTGAN) to generate positive samples, as shown in Figure 1. First, same as the original CGAN, the input of generator contains random noise z and a conditional vector c. Since the tabular data has high-dimensional features, a crosslayer is used to calculate the correlation between features before discriminator [26]. Stacking l-cross-layers can better model the relationship between variables and increase the variation of noise.

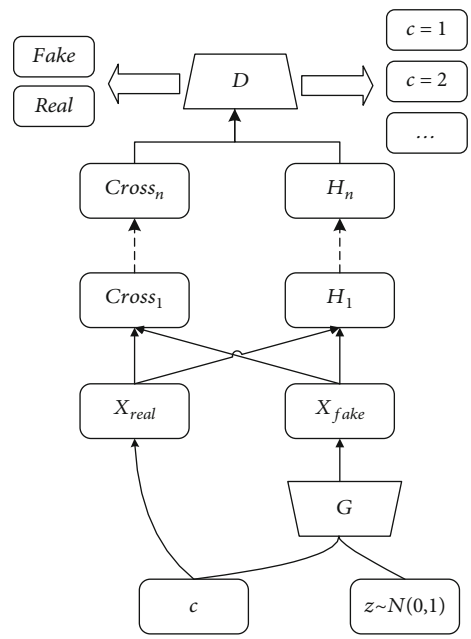


Figure 1: Structure of auxiliary conditional tabular GAN (ACTGAN).

In addition, an auxiliary classifier AC (Auxiliary Classifier) [27] is added to the discriminator D, so that the discriminator outputs both the true-or-false results of data and the classification result $\hat{c}$. The cross-entropy loss of AC is added to the discriminator loss. The output label $\hat{c}$ is compared with the generator input condition vector c to optimize the generator to generate samples that can identify a specific class. The formula is shown in (2), which calculates the binary cross-entropy loss between the true input label c and the predicted label $\hat{c}$, N represents the number of input samples.

$$L_{AC} = E_{c\sim p_{data}}\left[-\frac{1}{N}\sum_{i=1}^{N} c_i \bullet \log(\hat{c}_i) + ((1-c_i)\log(1-\hat{c}_i)\right] \tag{2}$$

In addition, since GAN has always been difficult to train, we use Wasserstein distance to replace the JS divergence in the original generative adversarial network [28–31]. Different from the gradient disappearance phenomenon of JS divergence and KL divergence, W distance has smooth characteristics and can be maximized by using a parameter value range, which can effectively narrow the generated distribution p_g and the real distribution p_{data}. The W distance formula is as follows:

$$W\left(p_g, p_{data}\right) = \frac{1}{K}\sup_{||f||\le K} E_{x\sim p_{data}(x)}(f(x)) - E_{\hat{x}\sim p_{\hat{x}}}(f(\hat{x})) \tag{3}$$

In formula (3), sup represents the least upper bound, and $||f|| \le K$ represents that the function f satisfies the

1-Lipschitz continuity, where the function f can be fitted by a neural network, so K is able to exist by restricting all parameters in the network not to exceed a certain range. Although weight clipping can make the network satisfy the 1-Lipschitz condition, the gradient will disappear due to the restricted weights and improper setting of weight clipping. Therefore, a gradient penalty (GP) is established for the above problem as shown in formula (4). The gradient penalty is a soft constraint, which can control the gradient around 1, alleviate the problem of gradient disappearance, and make the model stable.

$$L_{GP} = \lambda_{GP} E_{\hat{x}\sim p_{\hat{x}}}\left[\left(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1\right)^2\right] \tag{4}$$

We show the end-to-end training process of our data generation model ACTGAN in Algorithm 1:

(a) First, we manually specify network parameters $[\theta_g, \theta_d]$ and initialize generator G and discriminator D

(b) Sample real data x, condition vector c from real data; sample z from Gaussian-distributed noise data

(c) Generator G generates synthetic data X_{fake}

(d) Fuse real data with fake data as crosslayers' input, the crosslayer and the hidden layer calculate the high-dimensional feature interaction. After the neural network computing of the l_{th} layer, the embedding is obtained by concatenating hidden layers and crosslayers

(e) The embedding is input into the discriminator D to obtain the predicted true-or-false probability $\hat{y}$ and label value $\hat{c}$

(f) Set G constant, we optimize D through devised loss function L_D, in which $E_{x\sim p_{data}(x)}[D(x)] - E_{z\sim p_z(z)}[D(G(z|c))]$ calculating the binary cross entropy between real data and generated data. Auxiliary classifier loss L_{AC} is added so as to stabilize model training and make generated data closer to original ones. Moreover, L_{GP} is used to control the gradient from disappearing

We add The parameters of generator and discriminator are optimized by Adam optimizer, with learning rate $\eta = 0.0001$. The number of crosslayers l is set to 3.

3.2. ResNet-LSTM for Feature Extraction. The credit scoring data is usually a mixture of structured and semi-structured data, which are called multi-source heterogeneous data. We can divide the credit scoring data into two types: user profile data (i.e., gender, occupation and education) and time-based user behavior data (i.e., credit card transactions and previous loans data). Most researches only focus on a single type of data but fail to fuse these two types of data to extract high-level hidden feature. Some researches [32] treat all kinds of data equally and fail to capture the dynamics of user payment behavior over time, while others [33] only focus on user behavior data, not on user profile data that are critical to the credit scoring task. These conventional methods cannot mine and fuse the rich latent information from such multi-source heterogeneous credit data and thus fail to extract high-level hidden feature for credit scoring. In this context, the integration of multi-sources heterogeneous data has been considered as one of the crucial research points for credit scoring.

According to the above, after adding more positive samples to form an enhanced dataset, we propose a hybrid ResNet-LSTM model for the different structural features of the dataset, including two feature extractors, in which ResNet is used to extract static features, and the static features mainly include demographic characteristics (age, gender, height and weight, etc.), financial characteristics (income, property, etc.), dynamic characteristics include the user's operation sequence over a period of time (the number of applications per month, the number of repayments, etc.) and use LSTM to extract the entire business cycle. In order to capture the important information of the feature extraction object and suppress the irrelevant details, the spatial attention (SAM) module and the temporal attention (TAM) module are used to calculate the importance of static feature and dynamic feature vector, respectively, and A series of attention weight parameters are assigned to improve the network performance, and finally a fully connected layer is used for fusion and output to XGBoost to calculate the classification results. The framework flow chart is shown in Figure 2.

3.2.1. ResNet-LSTM. Static features contain a large amount of low-dimensional tabular data, which can be regarded as single-channel images. Convolutional neural networks have certain advantages in processing static image data. Operations such as convolution and pooling can reduce the amount of parameter calculation. Tabular data can be divided into multiple 1D vectors as the input of CNN, and the calculation amount of using 1-dimensional convolution in forward propagation is far less than the traditional CNN that uses images as input.

However, experiments show that the performance of traditional convolutional neural network will begin to degrade with the increase of network layers, which indicates that when the network becomes very deep, the depth network becomes difficult to train. This is because the ReLU function in the neural network only performs nonlinear transformation on the input data. If the low-dimensional data is redundant, huge amount of information will cause the mapping value of the neurons to flow through to be 0, that is, the gradient disappears. The residual network [34, 35] (ResNet) can solve the problem of performance declining. By short-circuiting the identity block and the ReLU layer through shortcuts can flexibly save useful information in the neural network and reduce the information redundancy in the data. Therefore the static features are trained with the residual shortcut structure.

At the same time, the consumer transaction data contains a large number of operation behavior sequences, which can be regarded as time series, and a sequence processor is

Require: The gradient penalty coefficient λ_{GP}, the number of iterations n_{critic}, the batch size m, Adam hyperparameters η
Create G, initialize θ_g, create D, initialize θ_d.
For $t = 1, \cdots, n_{critic}$ **do**
  For $i = 1, \cdots, m$ **do**
    Sample real data $x \sim p_{data}$, random noise $z \sim p(z)$, condition vector $c \sim p_{data}$.
$$X_{fake} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_m) = G(z|c)$$
    $Cross_1, H_1 = [X_{real}, X_{fake}]$
    $\cdots$
    $Cross_l = x_0 x_{l-1}^T \omega_{l-1} + b_{l-1} + x_{l-1}$
    $H_l = W_l x_{l-1} + b_l$
    $Embedding = \sigma([Cross_l^T, H_l^T] W_{logits})$
    $\hat{y}, \hat{c} = D(Embedding)$
    $L_D = E_{x \sim p_{data}(x)}[D(x)] - E_{z \sim p_z(z)}[D(G(z|c))] - L_{GP} + L_{AC}$
  End for
  Update parameter $\theta_d \longleftarrow \theta_d + \eta \nabla L_D(\theta_d)$
End for
  $L_G = L_{AC} - E_{z \sim p_z(z)}[D(G(z|c))]$
Update parameter $\theta_g \longleftarrow \theta_g + \eta \nabla L_G(\theta_g)$

Algorithm 1: ACTGAN training algorithm.

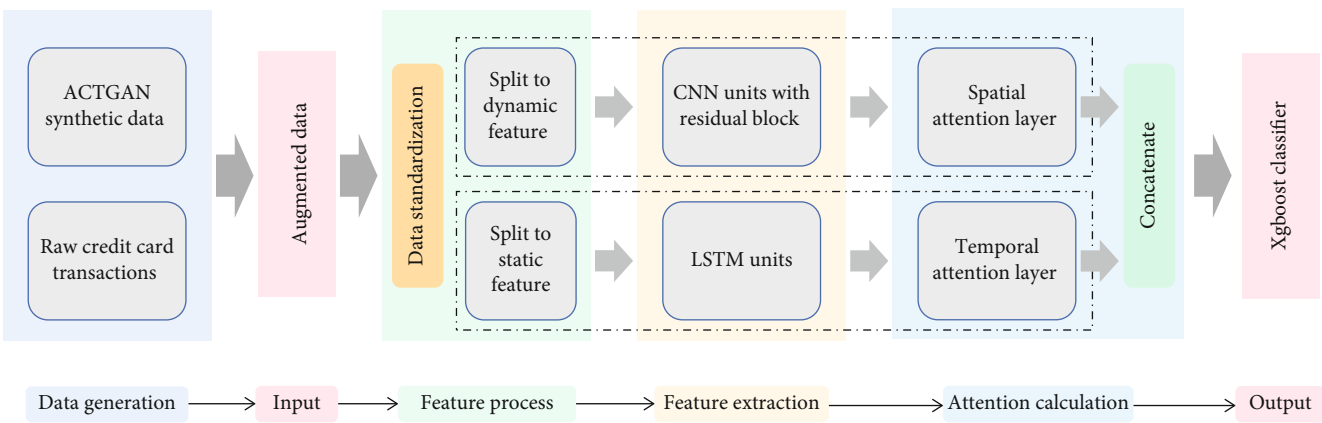


Figure 2: Credit scoring framework.

required to obtain the time embedding. Therefore, BiLSTM is used for modeling dynamic features. The two feature learners output the corresponding feature vectors, which are added to the attention mechanism module to improve model performance. The structure of ResNet-LSTM is shown in Figure 3.

3.2.2. Attention Module. Attention mechanism in the deep learning model mainly consists of two parts: first, determine which part of the entire input needs to be paid attention to; second, extract features from the key parts to obtain important information and sort the output. Attention mechanism can help the model assign different weights to each part of the input sequence, extract more key information and enable the model to make accurate judgments without bringing more computing and storage to the model. In the field of credit scoring literature [36, 37], the bidirectional LSTM network is used to add a temporal attention layer to predict credit card data, which has played a certain role in improving the classification effect of the model. However, these studies only used the temporal attention mechanism and did not consider the impact of nontime series data.

Moreover, different types of attention have different effects on the data. Since we input tabular data and some time series, the hybrid attention mechanism is used to capture importance information for different types of data. Tabular data can be regarded as a single-channel image and introduced into the image. Spatial attention is used to extract features, and time series uses the temporal attention mechanism to output the importance ranking of time points.

Spatial Attention Mechanism is proposed by the image field [38]. The process is shown in Figure 4. First, a global maximum pooling and global average pooling based on feature map X are performed to obtain average pooling feature map $AvgPool(X)$ and max pooling feature map $MaxPool(X)$, respectively, and then the two feature maps are concatenated to generate a valid feature representation vector. At last, through a one-dimensional convolution

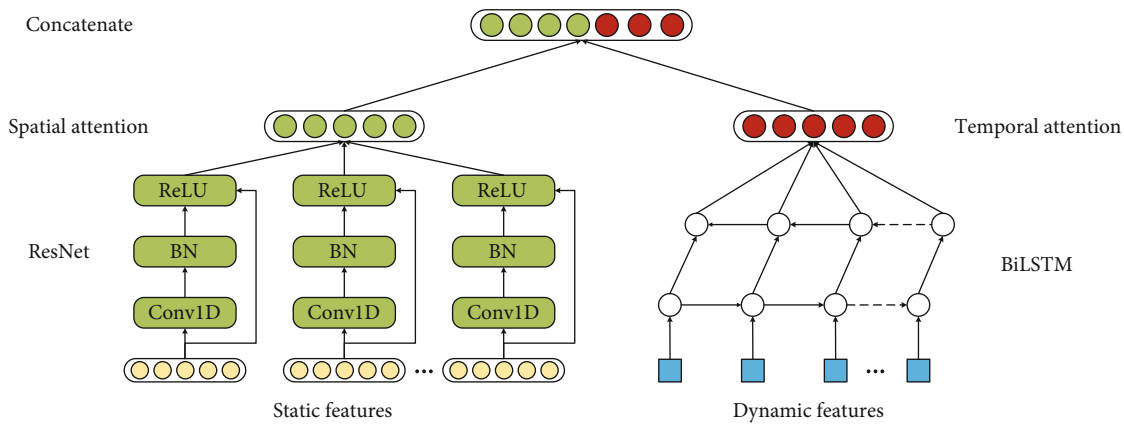


Figure 3: Structure of ResNet-LSTM.

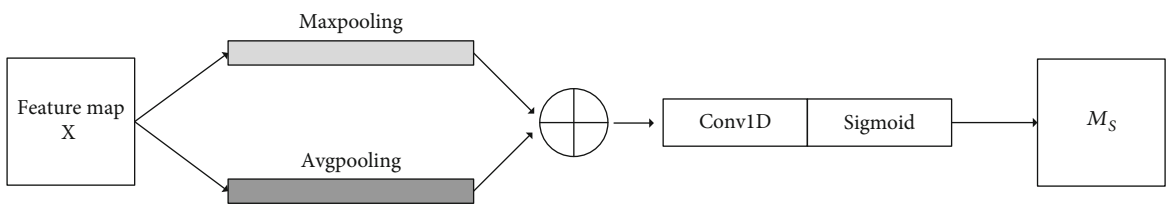


Figure 4: Spatial attention module.

operation f and a sigmoid activate function σ to generate a spatial attention feature vector, that is, M_s. The spatial attention formula is as follows:

$$M_s(X) = \sigma(f([AvgPool(X); MaxPool(X)])) \tag{5}$$

The input of the temporal Attention Mechanism is the hidden state of the BiLSTM at each moment. First, the input vector is encoded as a context vector by the tanh function, and the internal relationship of the time series data is learned by calculating the weight coefficient a, and the attention is allocated to the corresponding time to make the model focus on the more important subsequences. The temporal attention mechanism formula is as follows:

$$\begin{aligned} u_i &= \tanh\,(Wh_i + b) \\ a_i &= \text{softmax}(u_i) \\ M_D &= \sum_{i=1}^{t} a_i h_i \end{aligned} \tag{6}$$

Where h_i represents the output of the i time point of LSTM; t represents the length of the input sequence; a_i represents the weight of the output of the i time point; M_D refers to the weighted total of the LSTM output at each time point.

3.2.3. Focal Loss-XGBoost for Classification. Finally, we linearly fuse the two features learned by the neural network, and use the XGBoost classifier to output the final classification result $output_i$. Default customers are represented by 1, and normal customers are represented by 0. The formula is as follows:

$$\begin{aligned} X &= M_s \bigoplus M_D \\ output_i &= F(X) \end{aligned} \tag{7}$$

We use the Focal Loss function, which is the target detection algorithm, to optimize classification performance. The loss function specially designed for imbalanced classification is mainly added by the cross-entropy function with α coefficient and γ coefficient, so that the loss function is more inclined to focus on the minority class samples, so as to avoid the performance degradation caused by the easy-to-classify samples during the model training process. The formula is shown in (8), $\hat{y}$ represents the output probability of the classifier. When y=1, α is greater than 0.5 and less than 1 can increase the loss of misclassification, and the focusing coefficient γ can adjust the weight of easy-to-classify samples and hard-to-classify samples (γ >0). When y=1, the closer $\hat{y}$ is to 1, the smaller $(1-\hat{y})^{\gamma}$ is, it means that the sample is easier to be classified, the smaller the loss weight of the easy-to-classify sample is, thus make the classifier pay more attention to the hard-to-classify samples.

$$L_{FL} = \begin{cases} -\alpha(1-\hat{y})^{\gamma} \log \hat{y}, y = 1 \\ -(1-\alpha)\hat{y}^{\gamma} \log\,(1-\hat{y}), y = 0 \end{cases} \tag{8}$$

Table 1: Details of datasets used for evaluation.

Dataset	Normal samples	Default samples	Static features	Dynamic features	Imbalance rate
Bank	13307	2009	11	17	0.15
UCI	23364	6636	5	18	0.28

4. Experiments and Result Analysis

4.1. Dataset Description. We employ two real-world customer loan applicant datasets to implement and evaluate the proposed model, the first dataset is user loan data from an anonymous Chinese commercial bank, which contains about 15,000 consumer loan application records, including asset status, personal information, city of residence, equipment used, number of applications and other characteristics, nearly 15% of applicants are default users.

The second dataset, taken from the UCI Machine Learning Repository, is related to 30,000 applicants and transaction payments. It contains customer behavior data for the past 6 to 12 months (e.g. monthly/quarter/year application volume, billing amount, and default history), along with their financial and demographic information such as gender, city of work, age, property, and conditions. Nearly 22% of applicants are default users.

4.2. Metrics and Implementation Details. We divide the data into static features and dynamic features, and use the 0-1 label to indicate whether the customer defaults in the future. The details of the data set are shown in Table 1. We randomly split 70% of the data for training and 30% for testing. When training data, 20% of the data was randomly selected as the validation set. We adopt several metrics commonly used in credit scoring to evaluate the performance of the proposed model: AUC (Area under curve), recall, F1 value, and KS value.

TP (True Positive) indicates the actual default sample and the prediction is also a default, TN (True Negative) indicates that the actual nondefault sample is also predicted to be a nondefault, FP (False Positive) indicates that the actual default sample is predicted to be a nondefault, FN (False Negative) means that samples that are not in default are predicted to default. We calculate the true positive rate (TPR), false positive rate (FPR), F1 values and Recall values, the formula is as follows:

$$\begin{aligned} TPR &= \frac{TP}{TP+FN} \\ FPR &= \frac{FP}{TN+FP} \\ Recall &= \frac{TP}{TP+FN} \\ F1 &= \frac{2TP}{2TP+FP+FN} \end{aligned} \tag{9}$$

KS (Kolmogorov-Smirnov) is an evaluation index used to distinguish the degree of separation of positive and negative samples, and is often used in credit scoring models. The predicted outcome for each sample is a probability value in the range 0 to 1. The predicted probability values of positive and negative samples are arranged from smallest to largest, and the KS value is the absolute value of the largest difference between the two distributions. Generally speaking, the larger the KS value, the better the discrimination between positive and negative samples. The formula for the KS value is as follows:

$$KS = \max |TPR - FPR| \tag{10}$$

Table 2: Parameter setting of ResNet-LSTM.

Parameters	Value
batch_size	24/48
Convolution kernel size	16/32/64/128
LSTM units	128
Dropout	0.5
Activation function	ReLU
Optimizer/learning rate	Adam/0.0001
Epoch	100

The specific parameters of the neural network model are shown in Table 2. The convolution kernel is set to 4 layers of 16, 32, 64, and 128 units. In order to prevent overfitting, the drop out value is set to 0.5 to randomly inactivate 50% of the neurons. ReLU activation function and Adam optimizer to speed up convergence, learning rate is set to 0.0001.

4.3. Experimental Results

4.3.1. Imbalance Credit Scoring Result and Analysis. To verify the performance of ACTGAN, we train the model for 10000 iterations and compare with the vanilla conditional generative adversarial network (CGAN). As shown in Figure 5, the generator loss of the ACTGAN generator is lower than that of CGAN from the beginning, indicating that adding the crosslayer can well learn the interactive features of high-dimensional data, improve the ability of network feature extraction and enhance the performance of the generator. For the discriminator, the loss of ACTG AN after 8000 iterations is significantly lower than that of CGAN, which verifies that adding gradient penalty and auxiliary classifiers greatly improves the stability of model training. For W distance, CGAN stabilizes around 1 after 8000 iterations, compared with our proposed model, the Wasserstein distance of ACTGAN stabilizes around 0.6 and fluctuates slightly, indicating that ACTGAN has better convergence stability, and the synthetic data generated by

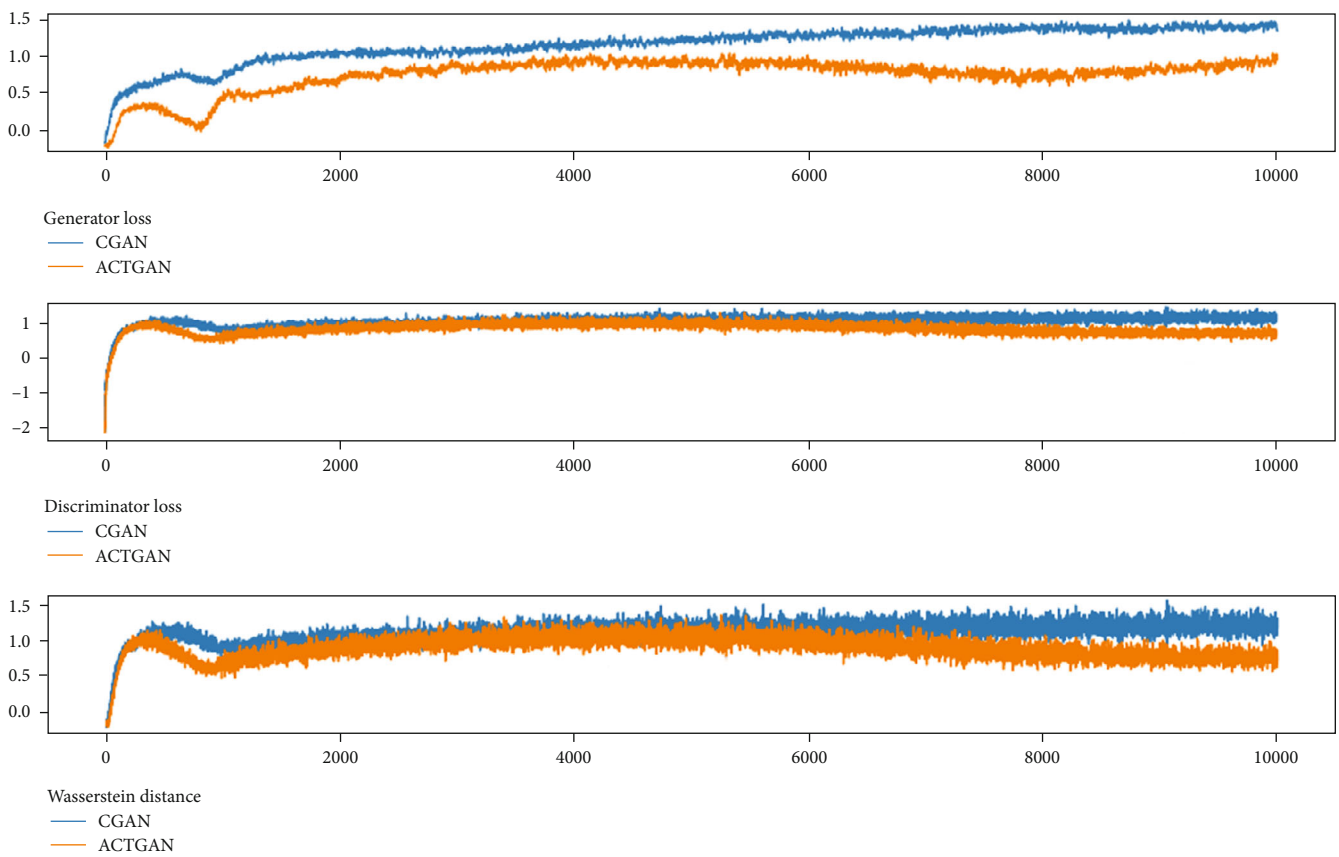


Figure 5: Comparison of training results.

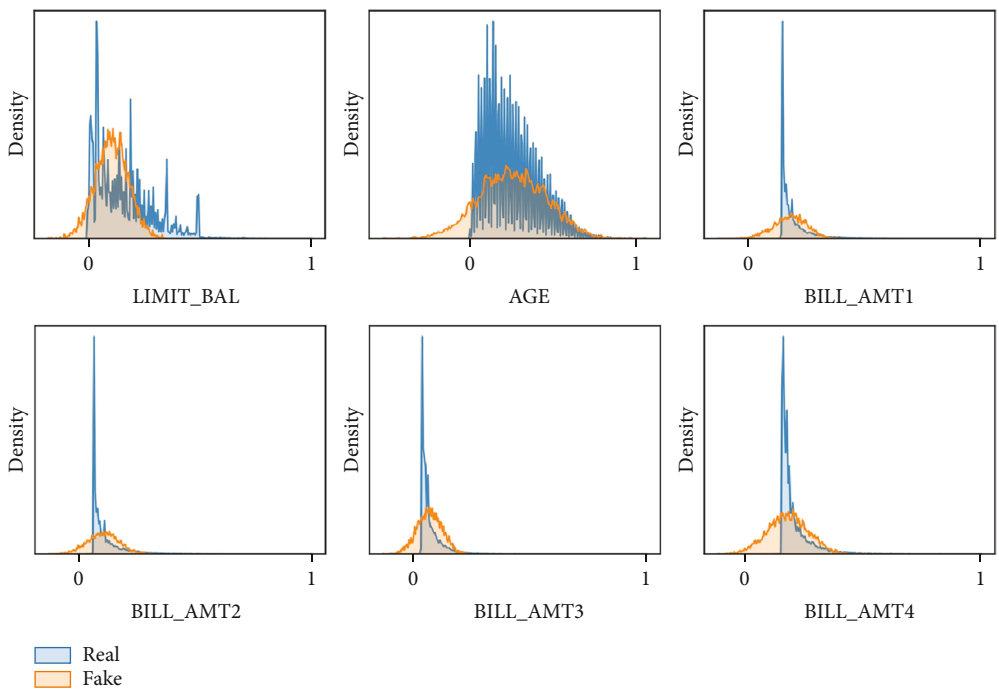
ACTGAN is distributed with a higher similarity to the real data.

Taking the six characteristics of age AGE, LIMIT_BAL, and BILL_AMT1-4 as examples, draw the distribution frequency map of real data and generated data. Since the data is preprocessed before being brought into the training model, the characteristics at this time can only reflect information, and does not represent the real attribute value. As shown in Figure 6, as the number of iterations increases, the data distribution generated by the generator G is getting closer and closer to the real data distribution, showing model's excellent learning ability.

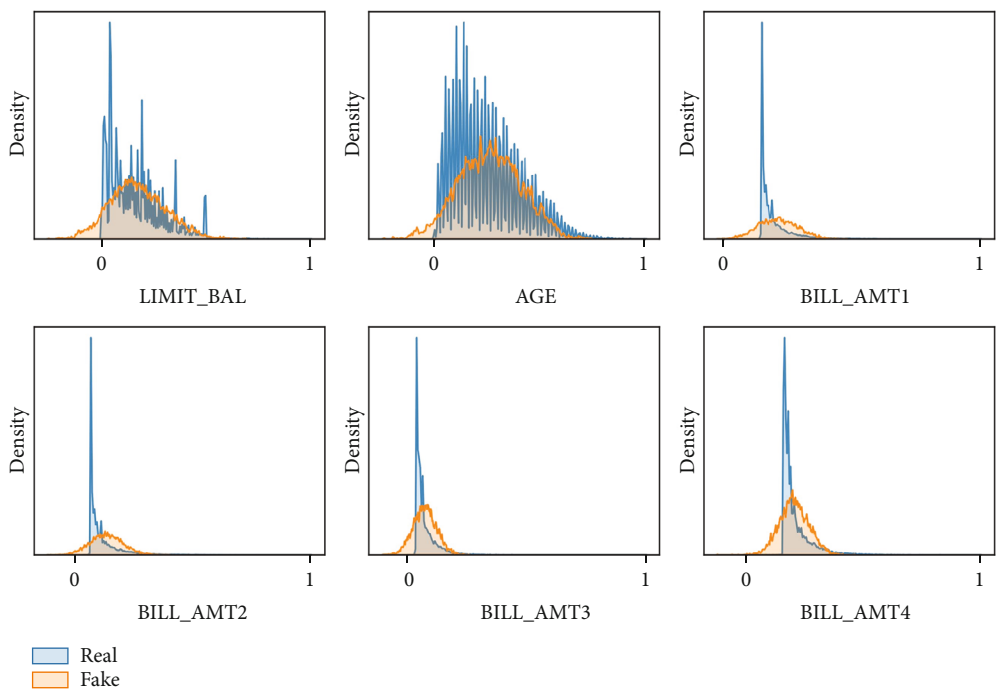
In order to reflect the enhancement effect of the data generation algorithm in this paper and the impact of data enhancement on the recognition performance of the classification algorithm, top data enhancement algorithms such as ADASYN, SMOTE, BorderlineSMOTE, and CGAN are selected to enhance the imbalanced data set in Table 1. Each dataset generates sufficient samples to increase the imbalance ratio to 0.4, and eventually forms an enhanced dataset; four machine learning algorithms: support vector machine, logistic regression, decision tree, and K-nearest neighbors are selected as classifiers, and the classifiers are compared in all performance metrics on the dataset.

In Tables 3 and 4, the bold font in each row represents the highest value of the row, from which it can be seen that the ACTGAN method is significantly better than other methods when using decision tree, support vector machine and k-nearest neighbor classifier for classification. Although the ACTGAN method does not show obvious advantages under some evaluation criteria, it may be related to the structural characteristics of some datasets. On the other hand, due to the training time of the ACTGAN model, the selection of the number of hidden layer nodes in the generator network and the discriminant network in the experiment is not very sufficient, and the number of training times of the model is not enough. The choice of hyperparameters are very dataset-dependent; but overall, the ACTG AN method still significantly outperforms several other resampling methods.

4.3.2. Ablation Study. Since we added an attention layer after the feature extraction layer, we performed ablation experiments on temporal attention and spatial attention. The results are shown in Table 5. It can be seen that the addition of the attention layer model has different degrees of improvement in the four evaluation indicators. In the UCI dataset, the effect of adding a single spatial attention SAM is better than that of adding a single TAM, and in the banking dataset, the effect of a single TAM is better than a single SAM, which may be due to the differences between the datasets and feature importance. Therefore, the combination of temporal and spatial attention can more comprehensively extract data feature information. The experimental results also show that the dual attention mechanism can effectively make up for the performance shortcomings of the single

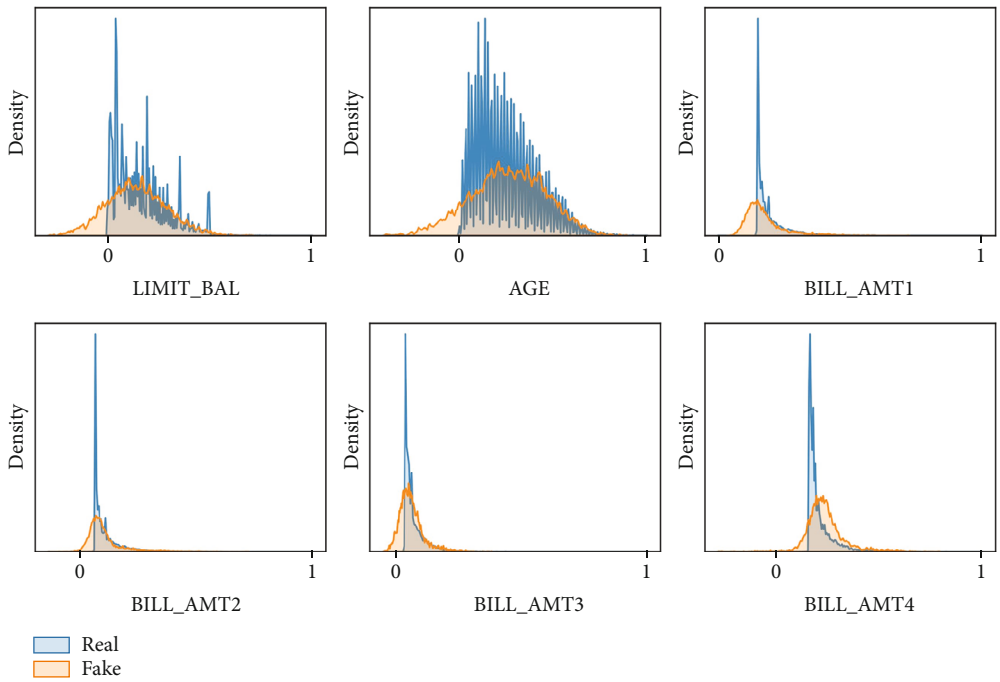


(a) 1000 iterations

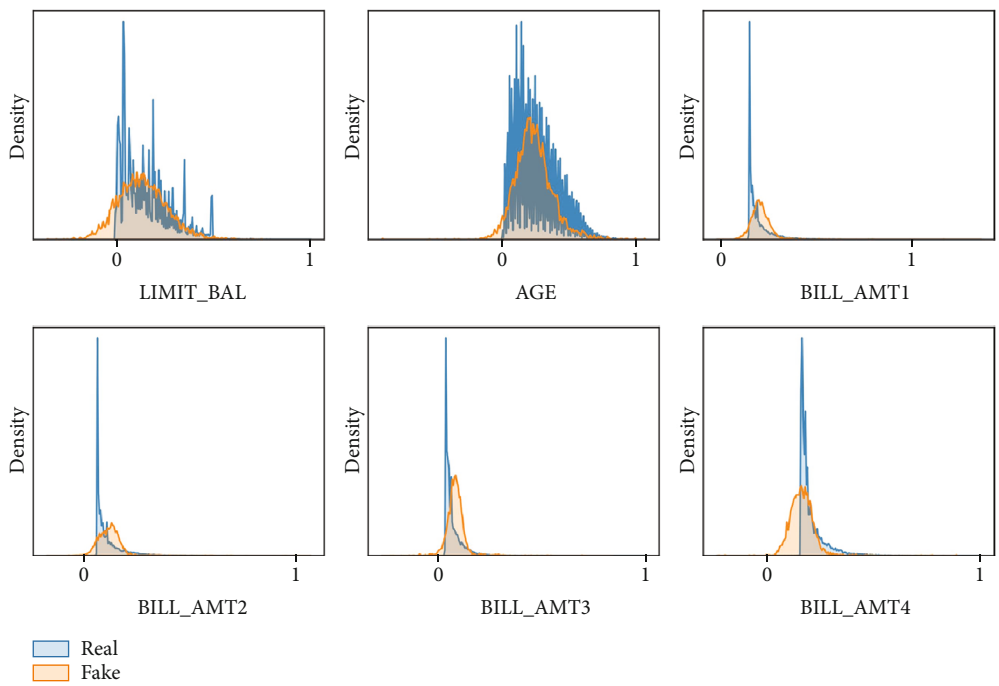


(b) 3000 iterations

Figure 6: Continued.



(c) 6000 iterations



(d) 9000 iterations

Figure 6: Data distribution of real data and fake data.

attention layer. The UCI data set applies the spatiotemporal attention module in the F1 value, AUC value is better than other methods. The banking dataset outperforms other methods in recall rate, F1 value, AUC value, and KS value, and the overall performance is the best.

4.3.3. Imbalance Classification Results. To demonstrate the advantages of the ResNet-LSTM+XGBoost algorithm, several baseline methods for imbalance classification are adopted for comparison. All hyperparameters are tuned on validation set with grid search.

RNN-RF [39]: RNN is used for dynamic temporal feature modeling, static features are extracted by feedforward neural network. Then the dynamic features and static features are fused and output to the random forest algorithm to predict the final default probability.

Table 3: Comparison of different data generation algorithm (Bank dataset).

	Critics	None	ADASYN	SMOTE	BorderlineSMOTE	CGAN	ACTGAN
Support vector machine	Recall	97.53	99.18	99.07	**99.25**	99.13	99.02
	F1	77.01	**90.66**	89.66	89.75	89.37	89.27
	AUC	94.52	95.38	95.11	95.21	95.56	**96.02**
	KS	88.41	84.59	82.90	83.25	83.48	**89.01**
Logistic regression	Recall	79.43	97.95	**98.75**	98.68	89.54	92.60
	F1	72.28	**90.31**	89.68	89.79	80.92	81.84
	AUC	86.50	94.91	95.05	**95.13**	90.37	92.40
	KS	70.01	82.98	82.46	82.72	84.65	**85.54**
Decision tree	Recall	62.99	86.59	85.12	86.11	69.24	**89.24**
	F1	63.20	**87.23**	85.17	85.42	71.05	81.05
	AUC	78.73	90.74	89.58	89.93	81.65	**91.34**
	KS	53.53	81.04	77.71	78.93	61.57	**81.69**
K nearest neighbors	Recall	24.70	90.22	88.98	89.48	94.46	**95.76**
	F1	13.45	**88.26**	87.11	87.33	84.95	84.05
	AUC	16.06	92.06	91.40	91.63	**94.42**	94.09
	KS	10.04	80.12	82.78	82.69	86.17	**87.46**

Table 4: Comparison of different data generation algorithm (UCI dataset).

	Critics	None	ADASYN	SMOTE	BorderlineSMOTE	CGAN	ACTGAN
Support vector machine	Recall	22.70	23.54	40.99	42.13	46.07	**47.32**
	F1	34.15	35.77	50.92	51.96	55.72	**56.98**
	AUC	59.92	60.36	66.47	67.04	69.16	**69.79**
	KS	19.70	20.09	32.12	33.37	37.75	**39.16**
Logistic regression	Recall	25.05	27.49	33.84	35.83	39.67	**43.13**
	F1	36.41	39.71	46.10	47.95	52.05	**55.17**
	AUC	60.78	61.83	64.31	65.18	67.27	**68.84**
	KS	21.52	23.67	26.99	28.83	32.57	**36.18**
Decision tree	Recall	41.68	51.73	56.17	55.95	55.63	**56.56**
	F1	39.92	49.75	55.19	54.93	**55.28**	55.19
	AUC	61.50	66.04	68.56	68.38	**68.64**	68.31
	KS	22.05	31.25	36.52	**36.95**	36.60	36.04
K nearest neighbors	Recall	34.08	41.14	46.46	**48.84**	44.65	46.49
	F1	41.79	47.99	53.83	**56.01**	54.64	55.84
	AUC	63.00	65.28	67.97	**69.28**	68.55	69.12
	KS	24.00	29.74	34.12	36.46	36.51	**37.89**

Table 5: Ablation study.

Method	Bank				UCI			
	Recall	F1	AUC	KS	Recall	F1	AUC	KS
None	90.33	81.23	91.45	85.41	55.24	65.52	74.86	52.85
SAM	89.35	80.31	90.84	84.38	57.60	66.11	75.37	52.70
TAM	94.24	82.22	93.07	84.87	54.51	65.24	74.66	53.58
SAM + TAM	**96.20**	**82.52**	**93.82**	**85.54**	57.32	**66.45**	**75.53**	53.49

Table 6: Classification method results of model.

Method	Bank				UCI			
	Recall	F1	AUC	KS	Recall	F1	AUC	KS
RNN-RF	82.07	78.44	87.89	70.64	25.77	35.60	60.24	20.28
Adaboost	**98.73**	78.89	95.25	87.25	31.89	43.55	63.92	27.66
SMOTEBoost	81.33	73.69	87.52	72.39	39.08	46.88	65.69	29.16
CUSBoost	74.05	70.22	84.09	64.50	36.33	46.11	65.21	27.59
RUSBoost	96.68	78.84	94.47	88.91	**61.43**	51.72	70.12	40.18
CNN	80.70	73.38	87.22	88.54	39.39	47.13	65.83	40.11
LSTM	95.09	79.03	93.91	89.46	28.01	38.10	61.36	34.05
Propose	98.41	**82.52**	**96.93**	**89.54**	57.34	**67.91**	**75.40**	**53.29**

Adaptive boosting tree (AdaBoost): iteratively optimize multiple weak classifiers and make them a strong classifier by adjusting the weights of misclassified data during each iteration.

SMOTEBoost [40]: Adaboost combined with random oversampling methods. SMOTE uses k-nearest neighbors to create synthetic examples of the minority class, then injects the SMOTE method on each boost iteration.

RUSBoost [41, 42]: compared to SMOTEBoost, RUSBoost achieves the same goal by performing random undersampling (RUS) at each boost iteration instead of SMOTE.

CUSBoost [43]: the majority class data is divided into K classes using the K-means algorithm (K is determined by hyperparameter optimization). Then within each cluster, randomly selected 50% of the data. Use the selected data together with the minority class data to form new balanced data.

CNN: The CNN-based architecture with a sliding window approach on behavioral data to overcome the class imbalance problem [44]. It has two convolutional layers which have filters of length 8 and 4. The number of the filters is set to be 32 and 64, and set the hidden layer dimension as 32.

LSTM: The LSTM model with hidden layer size of 128 for feature extraction.

The experimental results are shown in Table 6. The results show that the ResNet-LSTM +XGBoost model on the two datasets has obvious advantages over other algorithms in terms of F1, AUC value, and KS value. The recall rate of Adaboost on Bank dataset and the recall rate of RUSBoost on UCI dataset has exceeded our proposed method, this is due to differences in the distribution of minority class samples between datasets. Though the recall rate has not reach the highest score, the value of our proposed method is not far from the highest one. Meanwhile, in the other indicators our method has certain advantage, indicating that the ResNet-LSTM+XGBoost algorithm has excellent feature learning ability, and can extract information embedded in data nodes from different angles. Besides, improving Focal-XGBoost can also improve the classification performance of unbalanced learning.

5. Conclusion and Future Work

In order to solve the problem of data imbalance in the field of credit scoring, a novel data generation method ACTGAN for imbalanced data is proposed. Comparing results with other imbalanced data sample generation algorithms shows that the model training convergence speed and classification effect are improved. On this basis, a fusion deep neural network credit scoring framework based on ResNet-LSTM is proposed. In the framework, ResNet is used to extract static features from static financial data, and LSTM is used to extract dynamic features to detect the time dependence of user behavior data. And the spatio-temporal attention module is added into the framework to assign different weights to each unit when processing tabular data and time series to obtain detailed information of the focused target area. In the end, the Focal Loss function is introduced to improve the XGBoost classifier, thus the supervised learning ability of neural network is improved. We compare the performance of our hybrid deep learning model with other unbalanced classifiers proposed in related fields through experiments on the UCI dataset and private banking dataset, and the performance of our method has been significantly improved on F1 value, AUC, KS value.

For applications, this research provides compelling data and methodological support for more accurate solutions to credit scoring problems in the future, guiding the design of future methods for the same type of problems, proposing faster and more accurate solutions compared to previous solutions, and providing more applicability advantages in a future where data volumes and data dimensions are increasing. For the research object and domain, the study adds new insights into the optimization and application of ResNet-LSTM methods, which can be deepened in multiple contexts in the future, for example, in real-world problems such as geospatial assessment of inter-temporal evolution where data imbalances also exist. This research can be further improved in future study in several ways. First, in order to mine more information in the dimension of feature extraction and make full use of the user data provided by the credit platform, our future work will focus on social network data for entity relationship extraction, and try graph convolution networks to further extract data embedded in data nodes Information. Also, due to the small amount of dataset used in this study, we plan to incorporate more imbalance datasets to verify the reliability of our credit scoring model.

Data Availability

The data is available by sending email to corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Research and Development Project of the Ministry of Housing and Urban-Rural Development under Grant No. 2021-K-148, the Educational Commission of Zhejiang Province of China (No. Y202147553), and Zhejiang Provincial Natural Science Foundation of China (No. LGF20C050001).

References

[1] K. Bastani, E. Asgari, and H. Namavari, "Wide and deep learning for peer-to-peer lending," *Expert Systems with Applications*, vol. 134, pp. 209–224, 2019.

[2] M. O. Zan, G. A. I. Yanrong, and F. A. N. Guanlong, "Credit card fraud classification based on GAN-Ada boost-DT imbalanced classification algorithm," *Journal of Computer Applications*, vol. 39, no. 2, p. 618, 2019.

[3] Z. Z. SamanehSorournejad, R. E. Atani, and A. H. Monadjemi, *A survey of credit card fraud detection techniques: Data and Technique Oriented Perspective*, 2016.

[4] L. L. Song, S. H. Wang, C. Yang, and X. Sheng, "Application research of improved XGBoost in imbalanced data processing," *Computer Science*, vol. 47, no. 6, pp. 98–103, 2020.

[5] V. E. Neagoe, A. D. Ciotec, and G. S. Cucu, "Deep convolutional neural networks versus multilayer perceptron for financial prediction," in *2018 International Conference on Communications (COMM)*, pp. 201–206, Bucharest, Romania, 2018.

[6] L. Yu, R. Zhou, L. Tang, and R. Chen, "A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data," *Applied Soft Computing*, vol. 69, pp. 192–202, 2018.

[7] B. Zhu, W. Yang, H. Wang, and Y. Yuan, "A hybrid deep learning model for consumer credit scoring," in *2018 international conference on artificial intelligence and big data (ICAIBD)*, pp. 205–208, Chengdu, China, 2018.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[9] G. Douzas and F. Bacao, "Effective data generation for imbalanced learning using conditional generative adversarial networks," *Expert Systems with Applications*, vol. 91, pp. 464–471, 2018.

[10] U. Fiore, A. De Santis, F. Perla, P. Zanetti, and F. Palmieri, "Using generative adversarial networks for improving classification effectiveness in credit card fraud detection," *Information Sciences*, vol. 479, pp. 448–455, 2019.

[11] A. Blanco, R. Pino-Mejías, J. Lara, and S. Rayo, "Credit scoring models for the microfinance industry using neural networks: Evidence from Peru," *Expert Systems with Applications*, vol. 40, no. 1, pp. 356–364, 2013.

[12] N. Metawa, I. V. Pustokhina, D. A. Pustokhin, K. Shankar, and M. Elhoseny, "Computational intelligence-based financial crisis prediction model using feature subset selection with optimal deep belief network," *Big Data*, vol. 9, no. 2, pp. 100–115, 2021.

[13] S. Deng, R. Li, Y. Jin, and H. He, "Cnn-based feature cross and classifier for loan default prediction," in *2020 International Conference on image, video processing and artificial intelligence*, International Society for Optics and Photonics, 2020.

[14] C. Yan, X. Fu, W. Wu, S. Lu, and J. Wu, "Neural network based relation extraction of enterprises in credit risk management," in *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 1–6, Kyoto, Japan, 2019.

[15] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 1569–1572, Virtual Event China, 2020.

[16] B. Hu, Z. Zhang, C. Shi, J. Zhou, X. Li, and Y. Qi, "Cash-out user detection based on attributed heterogeneous information network with a hierarchical attention mechanism," *AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 946–953, 2019.

[17] Q. Zhong, Y. Liu, X. Ao et al., "Financial defaulter detection on online credit payment via multi-view attributed heterogeneous information network," in *Proceedings of The Web Conference 2020*, pp. 785–795, Taipei, Taiwan, 2020.

[18] M. A. Al-Shabi, "Credit card fraud detection using autoencoder model in unbalanced datasets," *Journal of Advances in Mathematics and Computer Science*, vol. 33, no. 5, pp. 1–16, 2019.

[19] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[20] A. Iranmehr, H. Masnadi-Shirazi, and N. Vasconcelos, "Cost-sensitive support vector machines," *Neurocomputing*, vol. 343, pp. 50–64, 2019.

[21] G. Paleologo, A. Elisseeff, and G. Antonini, "Subagging for credit scoring models," *European Journal of Operational Research*, vol. 201, no. 2, pp. 490–499, 2010.

[22] C. Luo, "A comparison analysis for credit scoring using bagging ensembles," *Expert Systems*, vol. 39, no. 2, article e12297, 2022.

[23] J. H. Wang and J. R. Yan, "Unbalanced data classification algorithm based on under-sampling and cost-sensitive," *Computer Applications*, vol. 41, no. 1, pp. 48–52, 2021.

[24] C. F. Tsai, Y. F. Hsu, and D. C. Yen, "A comparative study of classifier ensembles for bankruptcy prediction," *Applied Soft Computing*, vol. 24, pp. 977–984, 2014.

[25] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, https://arxiv.org/abs/1411.1784.

[26] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proceedings of the ADKDD'17*, pp. 1–7, New York, NY, USA, 2017.

[27] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *International conference on machine learning*, pp. 2642–2651, Sydney, NSW Australia, 2017.

[28] M. Zheng, T. Li, R. Zhu et al., "Conditional Wasserstein generative adversarial network-gradient penalty-based approach to

alleviating imbalanced data classification," *Information Sciences*, vol. 512, pp. 1009–1023, 2020.

[29] Y. Liu, Y. Zhou, X. Liu, F. Dong, C. Wang, and Z. Wang, "Wasserstein GAN-based small-sample augmentation for new-generation artificial intelligence: a case study of cancer-staging data in biology," *Engineering*, vol. 5, no. 1, pp. 156–163, 2019.

[30] I. Haloui, J. S. Gupta, and V. Feuillard, "Anomaly detection with Wasserstein GAN," 2018, https://arxiv.org/abs/1812.02463.

[31] X. Zhou, X. Xu, W. Liang, Z. Zeng, and Z. Yan, "Deep-Learning-Enhanced multitarget detection for end-edge-cloud surveillance in smart IoT," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12588–12596, 2021.

[32] P. R. Vardhani, Y. I. Priyadarshini, and Y. Narasimhulu, "CNN data mining algorithm for detecting credit card fraud," in *Soft Computing and Medical Bioinformatics*, pp. 85–93, Springer, Singapore, 2019.

[33] Y. Zhang, D. Wang, Y. Chen, H. Shang, and Q. Tian, "Credit risk assessment based on long short-term memory model," in *International Conference on Intelligent Computing*, pp. 700–712, Springer, Cham, 2017.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, pp. 770–778, Las Vegas, NV USA, 2016.

[35] W. Zheng, L. Yan, C. Gou, and F. Wang, "Federated meta-learning for fraudulent credit card detection," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 4654–4660, Yokohama, Japan, 2021.

[36] C. Wang, D. Han, Q. Liu, and S. Luo, "A deep learning approach for credit scoring of peer-to-peer lending using attention mechanism LSTM," *IEEE Access*, vol. 7, pp. 2161–2168, 2018.

[37] T. Liang, G. Zeng, Q. Zhong et al., "Credit risk and limits forecasting in e-commerce consumer lending service via multi-view-aware mixture-of-experts nets," in *14th ACM international conference on web search and data mining*, pp. 229–237, 2021.

[38] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, Cham, 2018.

[39] T. C. Hsu, S. T. Liou, Y. P. Wang, and Y. S. Huang, "Enhanced recurrent neural network for combining static and dynamic features for credit card default prediction," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1572–1576, Brighton, UK, 2019.

[40] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving Prediction of the Minority Class in Boosting," in *European Conference on Principles of Data Mining and Knowledge Discovery*, pp. 107–119, Springer, Berlin, Heidelberg, 2003.

[41] X. Zhou, X. Xu, W. Liang et al., "Intelligent small object detection for digital Twin in smart manufacturing With industrial Cyber-Physical Systems," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1377–1386, 2022.

[42] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: a hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, p. 185, 2009.

[43] F. Rayhan, S. Ahmed, A. Mahbub, R. Jani, S. Shatabda, and D. M. Farid, "Cusboost: cluster-based under-sampling with boosting for imbalanced classification," in *2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)*, pp. 1–5, Bengaluru, India, 2017.

[44] H. Kvamme, N. Sellereite, K. Aas, and S. Sjursen, "Predicting mortgage default using convolutional neural networks," *Expert Systems with Applications*, vol. 102, pp. 207–217, 2018.

Hindawi
Mobile Information Systems
Volume 2022, Article ID 8546576, 10 pages
https://doi.org/10.1155/2022/8546576

Research Article

Applying Machine Learning to Chemical Industry: A Self-Adaptive GA-BP Neural Network-Based Predictor of Gasoline Octane Number

Xingzhen Tao,[1] Yue Liu,[1] Haiping Li,[1] Yufei Xie,[1] Lin Peng,[1] Chao Li,[2] Lingling Guo,[3] and Yinling Zhang[3]

[1]*School of Information Engineering, Jiangxi College of Applied Technology, China*
[2]*Zhijiang College, Zhejiang University of Technology, China*
[3]*College of Chemical Engineering, Zhejiang University of Technology, China*

Correspondence should be addressed to Chao Li; cynthia0217@163.com

Received 18 February 2022; Revised 22 March 2022; Accepted 1 April 2022; Published 22 April 2022

Academic Editor: Yan Huang

Octane number is a measure of gasoline's ability to resist detonation and combustion in the cylinder; the higher the value, the better the resistance to detonation. The accurate prediction of octane loss during gasoline refining could facilitate production management and ensure gasoline octane. The backpropagation neural network is a traditional method adopted for the octane loss prediction, but there exists the issues of low training accuracy and poor generalization in the traditional BP neural network model caused by randomly generated weights and thresholds at input. In this paper, we propose a novel approach to optimize the weights and thresholds for gasoline octane number prediction based on a self-adaptive genetic algorithm. The experimental result shows that the proposed model outperforms in accuracy and generalization in the competition with the traditional BP neural network. The coefficient of determination R_2 of the performance index in the experiment is improved from 0.81502 to 0.95628, and the average prediction error among 10 groups of experiments was reduced from 0.0061 to 0.0041.

1. Introduction

Octane number is one of the most important properties of gasoline [1, 2], which directly affects the antiknock performance, fuel consumption, and low-temperature start and acceleration performances of automobiles. For product oil industry, gasoline octane number is an important quality index in the process of purchasing, storage, transportation, and sales [3]. Currently, the ASTM-CFR standard is the most commonly used standard octane test method; however, this method is expensive, and the test dosage is large, time-consuming, and complicated to operate [4]. In order to make up for the defects and shortcomings of experimental methods, theoretical prediction studies of octane number of gasolines can be carried out to establish a reliable prediction model since octane number of gasolines is closely related to its chemical composition [5]. In recent years, with the establishment of the Sinopec Sales Enterprise Laboratory Information Management System, the accumulation and sharing of quality data have been realized. Relying on the massive physical and chemical index data of gasoline in the database, it is possible to establish a gasoline octane number prediction model using machine learning algorithms [6]. Predictive models can divided into two categories [7]: one is linear models for predicting octane number, such as multiple linear regression analysis and partial least squares; the other is nonlinear models for predicting octane number, such as artificial neural network algorithms and support vector machine regression. A backpropagation (BP) neural network, one of the most reliable and classical neural networks among artificial neural networks, can be chosen as the base model with convenient operation and powerful learning ability [8–10]. However, the traditional BP neural network training suffers from slow convergence and low prediction

accuracy. To address these problems, a genetic algorithm is used to optimize the parameter training of the model [11–14].

In this paper, a BP neural network gasoline octane number prediction model is proposed based on self-adaptive genetic algorithm optimization with gasoline physical and chemical indexes as its independent variables and octane number as its dependent variable. Comparative experiments are further conducted to validate this model. The contribution in this work can be summarized as follows: (1) an optimized BP neural network model is proposed to address the issues existing in the traditional BP neural network while dealing with the task of gasoline octane number prediction and (2) a comprehensive experiment and analysis to validate the proposed method are carried out. The rest of this paper is organized as follows: the related work is introduced in Section 2, the proposed BP-based algorithm is detailed in Section 3, the experiment is detailed in Section 4, and Section 5 concludes the paper.

2. Related Work

2.1. Gasoline Octane Number Prediction. Octane number is a measure of the gasoline's ability to resist detonation in the cylinder; the higher the value, the better the resistance to detonation. The quantitative analysis models of octane value are divided into linear and nonlinear models; linear models include multivariate statistical analysis methods [15], Raman spectral data combined with the partial least squares regression (PLS) algorithm [13], Raman spectrometer combined with the PLS algorithm [16], momentary combined with the local PLS algorithm (MC-PLS) [14], Fourier transform spectroscopy combined with PLS [17], linear predictive coding combined with MLR [18], NIR spectroscopy based on ANN, support vector machine (SVM) and multivariate statistical analysis [19], and the use of NIR spectroscopy. Most of the nonlinear models used algorithms such as multiple linear regression (MLR), Principal Component Regression (PCR), and ANN [20], among which ANN is the most effective; [21] used short-wave NIR spectra with laser-induced spectroscopy to develop an octane analysis model. Quantitative octane analysis requires a large sample set and a high level of model complexity. Linear and nonlinear models are slightly simpler to construct compared to linear models but are not as accurate as building complex nonlinear models. The nonlinear model requires higher sample set capacity and depends on the optimization of parameters and extraction of features.

The prediction of quality indicators such as gasoline octane number is usually modeled by combining nonlinear models with intelligent algorithms. Since the actual octane detection process often has the disadvantages of slow detection speed and large amount of pollutants emitted, NIR spectroscopy gradually becomes the mainstream octane detection method. The BP neural network is a very mature multilayer feedforward network, which mainly contains three parts: input, hidden, and output layers, and it is very sensitive to the initial weights and has different convergence speeds when given different initial values. The key to detecting octane values by NIR spectroscopy is to build a mathematical model, and BP neural networks are widely used in octane prediction problems because of their strong generalization, self-adaptive capability, and ability to approximate any nonlinear connectivity function with arbitrary accuracy. Octane loss modeling using BP begins with preprocessing the data to filter out some of the variables with the highest correlation to octane values. The number of neurons of the network directly affects the prediction results, and the BP neural network is sensitive to the initial parameters, and some optimization algorithms have been proposed by scholars in order to accelerate its convergence [4, 22–25].

2.2. BP Neural Network. The BP neural network was firstly proposed by Rumelhart and McClelland in 1986 [26]. This algorithm mainly includes two calculation processes [15, 27–29]. The first is to propagate the output error through the direction from input to output and, at the same time, to continuously adjust the weights and thresholds according to the training objectives of the network. If the actual output is not consistent with the expected output, it is necessary to switch to the second calculation stage, i.e., the error backpropagation process. In the second process, the input layer is retransmitted layer by layer to decrease the error, which adjusts parameters along the gradient direction. Through learning and training these two processes repeatedly, the network weight and threshold corresponding to the minimum error are determined, and the network model is created, leading to the end of the model. The algorithm pseudo-code is described in Algorithm 1.

The output of a neuron j on the output and hidden layers of the BP neural network is formulated as follows:

$$O_j = f_j\left(\mathrm{Net}_j\right) = f_j\left(\sum_{i=1}^{l} w_{ij}x_i + b_j\right), \tag{1}$$

where x_i denotes the individual input values of neuron j, O_j denotes the output value of neuron j, w_{ij} denotes the individual connection weights between the corresponding input i and neuron j, f_j denotes the activation function of neuron j, the sigmoid function $y = 1/(1 + e^{-x})$ is commonly used as the activation function, and b_j denotes the threshold value of neuron j.

The commonly used empirical formula for the number of implicit layers is

$$h = \sqrt{m + n} + a. \tag{2}$$

Also, the loss function of the error is

$$F(w, b) = \frac{1}{2}\sum_{i=1}^{M}\left(d_i - O_i(w, b)\right)^2, \tag{3}$$

```
BPBtrain(){
   Initialize network"s weights and thresholds;
While termination conditions are not met {
   For each training sample X in the samples {
// Forward propagation of inputs
For Each cell j of the hidden or output layer {
         I_j ∑_j w_ij O_j + θ_j ; ;
            // The net input to the computational cell is relative to the previous layer i
         O_j = 1/(1 + e^(−I_j));
            // Calculate the output of cell j. Choose the sigmod function as the activation function
         }// Reverse propagation error
   For each cell j of the output layer{
         Err_j = O_j(1 − O_j)(T_j − O_j);// Calculate error
}
   From the last to the first hidden layer, for each cell j of the hidden layer{
               Err_j = O_j(1 − O_j) ∑_k Err_k w_kj;
               // k is a neuron in the next layer of j
               }
   For each weight wij in the network {
         ΔW_ij(l) Err_j O_i;
            // weighted value added, where is the learning rate
         W_ij = W_ij + ΔW_ij;
            // weight update
         }
   For each deviation in the network {
         Δθ_j = (l) Err_ij// Value added deviation
         θ_j = θ_j + Δθ_j;// Deviation update
         }
      }
   }
}
```

Algorithm 1: BP neural network training algorithms.

where w and b denote the weight and threshold, respectively; M is the number of samples; d_i denotes the expected result of the i_{th} sample; O_i records the actual network output of the i_{th} sample; and $f(w, b)$ is the loss function.

2.3. Genetic Algorithm. In 1970, Professor Holland proposed the genetic algorithm, which is a self-adaptive global optimization search algorithm. This algorithm is efficient, practical, and robust and has been widely used in different fields [30–33], such as machine learning, pattern recognition, neural networks, control system optimization, and social sciences [34–38], by repeating three key operations on the current population, that is, selection, crossover, and mutation. To help the population gradually evolve to a state close to the optimal solution, it uses population search techniques, which can be realized through repeating three key operations (i.e., selection, crossover, and mutation) on the current population [39–43]. The genetic algorithm is an encoding of the problem parameters to be optimized, and its basic operations are as follows:

(i) The population initialization: randomly generate N individuals as initialized populations to ensure sufficient diversity in the population

(ii) The fitness function: it is the criterion to distinguish the good and bad individuals in the population, and the direction of increasing fitness function is in the same direction as the change of the genetic algorithm

(iii) Selection arithmetic: application of the roulette wheel selection method to select chromosomes into the next generation according to their cumulative probability

(iv) Crossover operation: randomly selecting two chromosomes and then generating a random number to produce two new individuals, which is crossed over if it is less than the crossover rate

(v) The mutation operation randomly selects a chromosome and generates a random number, which mutates if it is less than the mutation rate, and then, a new individual is produced

(vi) Population update: the selected solutions by genetic manipulation are saved and finally an optimal population is obtained

The pseudo-code description of the genetic algorithm [34] is shown in Algorithm 2.

```
Parents < - { Randomly generated populations }
   While not (Termination condition)
      Calculate the fitness of each parent in the population
Children < - ∅
   While | Children| < |Parents|
      Using fitness to select a mating pair of sires based on probability
         The parents mated to produce offspring c1 and c2
            Children < - Children{c1, c2}
   Loop
      Some offspring random mutation
Parents < - Children
   Next Generation
```

Algorithm 2: Genetic algorithm.

3. Optimized Neural Network Model

3.1. Establishment of the Neural Network Model

3.1.1. Building the BP Neural Network Model. The gasoline octane number rationalization index contains 401 feature data; thus, the input port neurons of the neural network are 401 and the output port neuron is 1. According to the best test results of the empirical formula, the number of neurons in the hidden layer is set at 25. The neural network model is constructed as shown in Figure 1, and the sigmoid function is used for the activation function of the hidden layer and the output layer.

There are still some shortcomings in the BP neural network: (1) the learning rate of the neural network is determined by experimental experience, and it is difficult to find the optimal value and (2) the initial weights and thresholds of the neural network are randomly generated, which is easy to obtain the local optimal values and affects the model prediction performance. Therefore, the following improvements are made to address these problematic topics: (1) the learning rate is no longer set at a fixed rate, and the self-adaptive learning rate is used to improve the learning efficiency of the network and (2) the genetic algorithm is very effective in finding optimal values, so it can be used to find the optimal weights and thresholds, preventing the local optima and improving the model prediction effect.

3.1.2. Learning Rate Optimization Based on the Self-Adaptive Algorithm. Traditional neural networks are trained with a fixed learning rate, which has a great impact on the training results. Unfortunately, it is uncertain about the learning rate. If the learning rate is too small, the more training times are required and the network converges are slower, while if the learning rate is too large, the stability of the network structure is poor. Therefore, it is important to examine the method of the self-adaptive learning rate. The formula is as follows:

$$\gamma_{t+1} = \begin{cases} e^{0.002} * \gamma_t, & \Delta E < 0, \\ e^{-0.002} * \gamma_t, & \Delta E \geq 0, \end{cases} \tag{4}$$

where y_t is the learning rate used at the t_{th} training, γ_{t+1} is the learning rate used at $(t+1)_{\text{th}}$, and Δ_E is the amount of variation in error.

3.2. BP Neural Network Model Optimized by the Genetic Algorithm

3.2.1. Improvement of the Genetic Algorithm

(1) Initial population and individual codes. N individuals are randomly generated as the initial population, considering that the number of input nodes of the neural network is m, the number of nodes of the hidden layer is h, and the number of nodes of the output layer is n. The following issues should be noted when coding chromosomes: (1) the weight matrix is a two-dimensional matrix, while chromosomes are one-dimensional; (2) there are multiple weights and thresholds, but a chromosome is fixed. Thus, the two-dimensional matrix of weights is mapped into a one-dimensional matrix, and multiple weights and thresholds are spliced into chromosomes. Using real number encoding, each chromosome is actually a string of real numbers, which consists of connected weights and thresholds for each layer of the network model. The formula to determine the length of individual chromosomes is as follows:

$$L = m * h + n * h + n + h. \tag{5}$$

(2) Fitness function. The fitness function is a criterion to distinguish the good and bad individuals in the population. For the randomly generated weights and thresholds, the resulting error is calculated, and the direction of the increasing fitness function follows the same direction of the genetic algorithm evolution. Here, the inverse of the loss function is chosen as the fitness function:

$$\text{Fit} = \frac{1}{1/2 \sum_{i=1}^{M} \left(d_i - O_j\right)^2}, \tag{6}$$

where M denotes the number of training set samples, d_i denotes the expected output of the i_{th} training sample

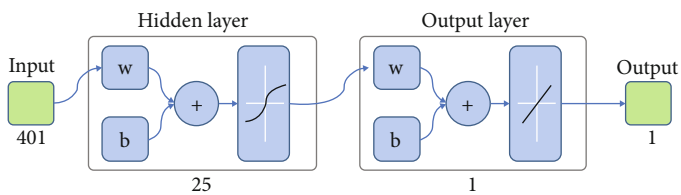


Figure 1: The structure of the neural network model.

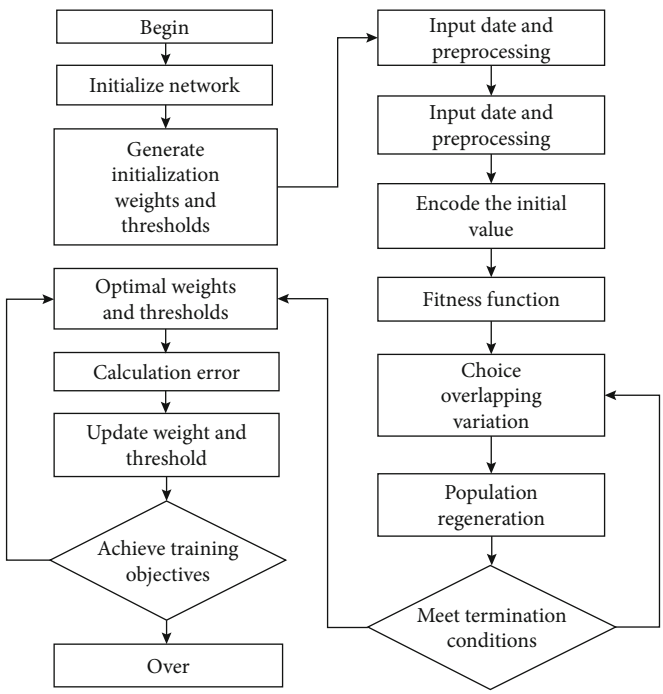


Figure 2: The flow chart of the specific GA-BP network model.

network, and O_i denotes the actual output of the i_{th} training sample network.

(3) Selection operation. The selection operation uses the roulette wheel selection method, whereby chromosomes are selected to produce populations with the same number of individuals as N populations. In the selection process, there may be duplicate individuals, and duplicate individuals are irrelevant when crossover is performed. Therefore, duplicate individuals are also eliminated during the selection process.

(4) Crossover operations. The crossover operation uses a single-point crossover, in which two paired individuals are selected from the initial population. During the process, a random crossover point is set, and parts of the chromosomes are swapped with the formation of two new individuals.

The purpose of the crossover operation is to generate new population individuals to improve the population diversity, and thus, the value of the crossover rate is of great importance to the performance of the genetic algorithm. Generally, the standard genetic algorithm uses a fixed crossover rate, which leads to problems such as premature algorithm or slow convergence. If the crossover rate is too small, it is difficult for the population to produce good individuals. If the crossover rate is too large, it is difficult to retain the good individuals in the population in the later stage of the algorithm. Therefore, the crossover rate used in this paper is varied with the change of fitness value, and the formula is as follows:

$$P_C = \begin{cases} P_{c1} - \dfrac{(P_{c1} - P_{c2})\left(f' - f_{\text{avg}}\right)}{\text{f}_{\max} - \text{f}_{\text{avg}}}, f' \geq f_{\text{avg}}, \\ P_{c1}, f, \end{cases} \tag{7}$$

where $P_{c1} = 0.99$, $P_{c2} = 0.4$, $f_{\max}$ is the largest fitness value in the population, f_{avg} is the average fitness value of the population per generation, and f' is the larger fitness value of the two individuals that will cross over.

(5) Mutation operations. Due to the long length of the chromosome, it is not suitable to choose the traditional single-point mutation operation. The number of mutation sites

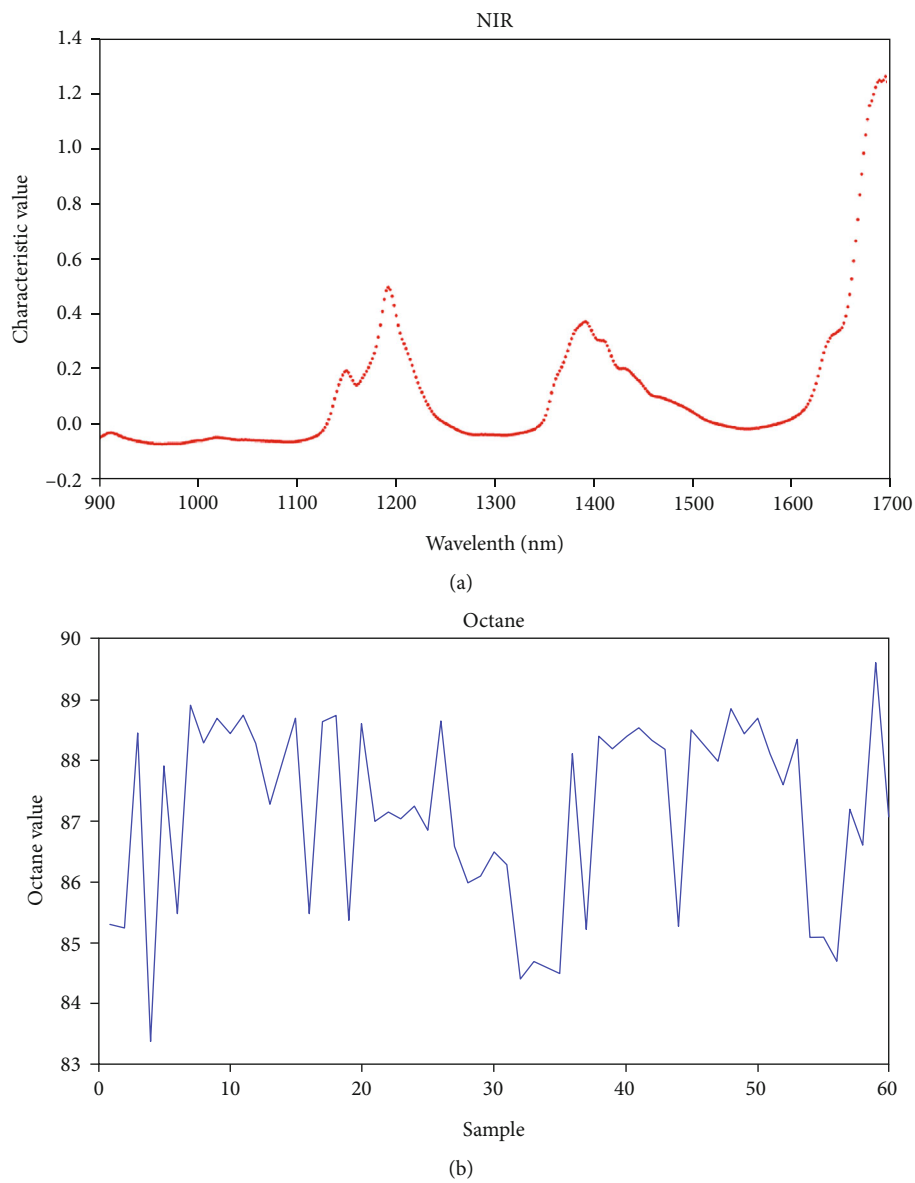


Figure 3: The dataset (a) shows the NIR spectral analysis results of one group of gasoline samples and (b) shows the octane number of 60 groups of gasoline samples.

changes, and the method of self-adaptive mutation sites is used with the following equation:

$$L = L\left[L_{\max}\left(1 - \frac{G}{G_{\max}}\right)\right]. \tag{8}$$

When the mutation operator mutates the k_{th} gene of an individual a with a certain probability. The mutation operation used is as follows:

$$a_k{}' = \begin{cases} a_k + (a_{\max} - a_k) * r_1\left(1 - \dfrac{G}{G_{\max}}\right)^2 r_2 > 0.5, \\ a_k + (a_{\min} - a_k) * r_1\left(1 - \dfrac{G}{G_{\max}}\right)^2 r_2 \leq 0.5, \end{cases} \tag{9}$$

where a_k denotes a mutation in the k_{th} gene of an individual a, $a_k{}'$ is the mutated individual, $a_{\max}$ and $a_{\min}$ are the upper and lower bounds of individual gene values, G is the current number of iterations, $G_{\max}$ is the maximum number of iterations, r_1 and r_2 are random numbers between $[0, 1]$, and $L_{\max}$ is the predetermined maximum number of mutated bits. The advantages of using this variation operation are as follows: (1) the setting of the random number, r_1, can influence the degree of variation; (2) the setting of the random number, r_2, can ensure that the gene value increases or decreases with equal probability, while the existence of the upper and lower bounds of the gene value ensures an appropriate variation of the gene value; and (3) the self-adaptive module used to adjust the number of variance bits and take into account the balanced search ability of the algorithm in both global and local. The degree

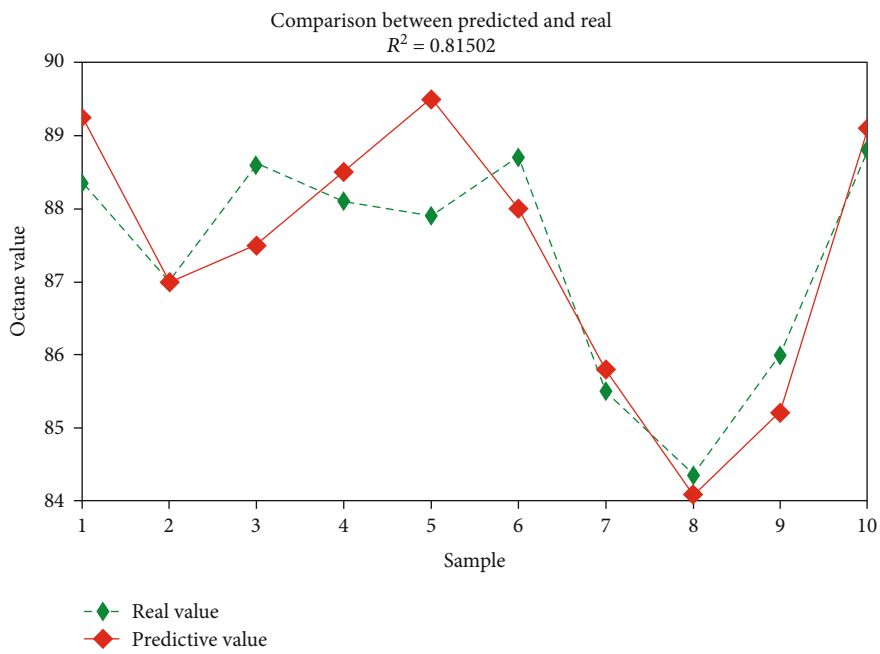


Figure 4: The results of the predicted values were compared with the real gasoline octane number in the BP model.

of variance decreases gradually with the increase in iterations, which ensures the strong global search ability at the beginning and the local search ability of the algorithm at the later stage. It can greatly prompt the individual to converge to the global optimal solution.

3.2.2. Optimization of Weights and Thresholds for the BP Neural Network. The processes of building a BP neural network (as described in Algorithm 1) based on the self-adaptive genetic algorithm (as described in Algorithm 2) optimization are as follows:

(1) Creating a BP neural network, randomly generating initial weights and thresholds, initializing populations, coding with real numbers, and determining the number of populations

(2) Calculating the fitness function and selecting the best individuals according to the roulette selection method and inserting them into the next-generation population

(3) Generating new individuals by crossover and mutation in the new generation of populations

(4) Reinserting new individuals into the population and calculating their fitness values

(5) Terminating the algorithm if a satisfactory individual can be found; otherwise, go back to step (2)

(6) After finding the optimal individual, the individual is decoded to obtain the optimized weights and thresholds, which are then used in the BP neural network

The flow chart of the specific GA-BP network model is shown in Figure 2.

4. Experiment

4.1. Experimental Dataset. In the experiment, 60 groups of gasoline samples were selected and analyzed by Fourier near infrared spectroscopy (900-1700 nm). The wavelength point was taken as an eigenvalue at an interval of 2 nm, and 401 eigenvalues were obtained to form the dataset (i.e., spectra_data.mat), containing two sets of values, matrix NIR and matrix octane. Among them, NIR stores the physicochemical data of gasoline collected by infrared spectroscopy. Octane stores the real octane number corresponding to these 60 eigenvalues. Figure 3(a) shows the NIR spectral analysis result of one specific group of gasoline samples, and Figure 3(b) shows the octane number of a total of 60 groups of gasoline samples.

In this experiment, the dataset is randomly split into two parts, in which $50 * 401$ data are used as the training set and the other $10 * 401$ data are used as the test set. Since the order magnitude of each feature data of gasoline octane number is inconsistent, this will affect the final mapping results. The dataset needs to be normalized by the following formula:

$$t = \frac{t - t_{\min}}{t_{\max} - t_{\min}}, \tag{10}$$

where $t_{\min}$ is the minimum value for each column of data, $t_{\max}$ is the maximum value for each column of data, and t is the value to be normalized.

Further, the performance evaluation of the gasoline octane number prediction model is divided into two parts: the relative error E and the coefficient of determination R_2, which are defined as follows:

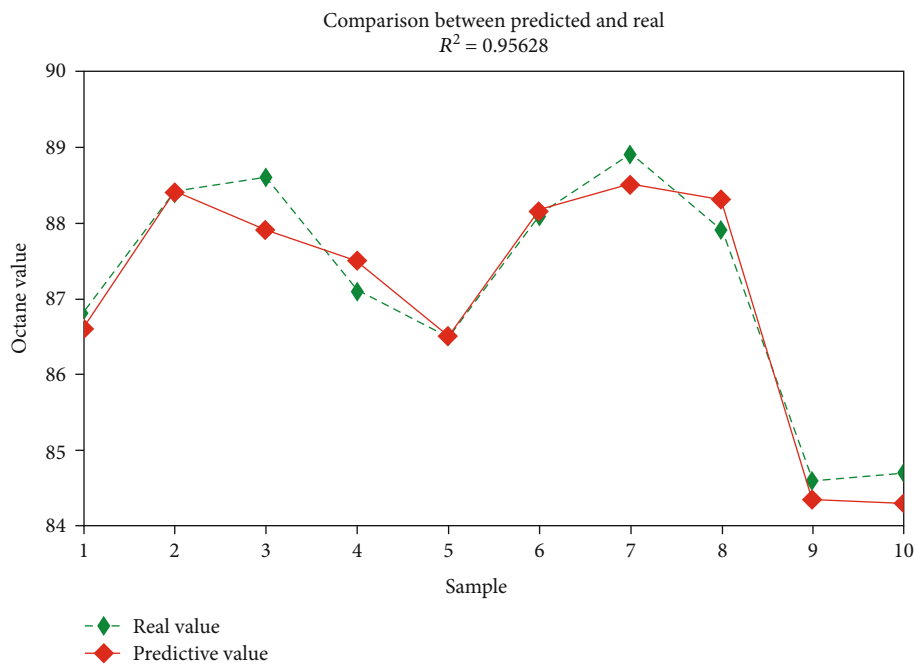


Figure 5: The results of the predicted values were compared with the real gasoline octane number in the GA-BP model.

$$E_i = \frac{|\hat{y}_i - y_i|}{y_i} \ (\mathrm{i}, 2 \cdots, \mathrm{n}),$$

$$R_2 = \frac{\left(l\sum_{i=1}^{l}\hat{y}_i y_i - \sum_{i=1}^{l}\hat{y}_i \sum_{i=1}^{l} y_i\right)^2}{\left(l\sum_{i=1}^{l}\hat{y}_i^2 - \left(\sum_{i=1}^{l}\hat{y}_i\right)^2\right)\left(l\sum_{i=1}^{l}y_i^2 - \left(\sum_{i=1}^{l}y_i\right)^2\right)}, \tag{11}$$

where $\hat{y}_i$ is the predicted value of the i_{th} sample, y_i is the true value of the i_{th} sample, and n is the number of samples. It is clear that the smaller the relative error, the better the performance of the model. The closer the coefficient of determination to 1 in the range of [0,1], the better the performance of the model. Conversely, the closer the coefficient of determination tends to 0, the worse the performance of the model.

4.2. Experimental Results and Simulation Analysis. The BP neural network model uses the BP network described in Section 3.1, with an initial learning rate of $\lambda = 0.1$, a maximum number of iterations set to 2000, and a minimum acceptable error set to 0.001. The training set was first applied to train the constructed model, and then, the validation test of the model was completed with the test set. Finally, the predicted values of the model were compared with the real gasoline octane number in the dataset, and the results were obtained as shown in Figure 4. The model performance index R^2 was 0.81502, which shows that the BP neural network model can predict the gasoline octane number. Unfortunately, the limitations of the traditional model lead to a low accuracy of the prediction.

Table 1: The comparison of the relative errors produced by 10 predictions for the two models.

No.	BP	GA-BP
1	0.0073	0.0035
2	0.0092	0.0061
3	0.0108	0.0040
4	0.0031	0.0024
5	0.0026	0.0071
6	0.0071	0.0035
7	0.0030	0.0065
8	0.0064	0.0045
9	0.0057	0.0034
10	0.0055	0.0063

Based on the optimized BP neural network based on the genetic algorithm designed in Section 3.2, the gasoline octane number GA-BP model is established. In the genetic algorithm, the population size is set to 100, and the maximum number of iterations is 500. In this model, the crossover rate is varied by self-adaptive, the variation rate is 0.09, the upper bound of the gene value is 1, the lower bound of the gene value is -1, and the maximum number of variation bits is set to 10. The same comparative experiments were used, and the experimental results are shown in Figure 5. The model performance index R_2 was 0.95628. The results show that the accuracy of the optimized BP neural network for predicting the gasoline octane number is improved by 14%.

In order to obtain accurate judgment of the experimental results, the evaluation index was the average error of the loss function through multiple experiments, resulting from the

randomness of some parameters in the experiments. Each set of data was summed and averaged the loss values through the loss function. The data in Table 1 shows the comparison of the relative errors produced by 10 predictions for the two models, respectively.

It is notable in Table 1 that the average value of the relative error is 0.0061 in the traditional BP neural network, while the average value of the relative error of the BP neural network optimized by the self-adaptive genetic algorithm is 0.0047, which reduces the error value by 23%.

5. Conclusion

Octane number is an important metric describing gasoline's ability to resist detonation and combustion in the cylinder; the higher the value, the better the resistance to detonation. The accurate prediction of octane loss during gasoline refining could facilitate production management and ensure gasoline octane. A neural network, which is with excellent performance in dealing with nonlinear system problems, is widely used in a number of fields. However, there are still some deficiencies. In this paper, we optimize the weights and thresholds of the neural network by the self-adaptive genetic algorithm and self-adaptively adjust the learning rate to improve the accuracy and generalization ability of the model. A novel GA-BP model was established, and this model was used for gasoline octane number prediction. Through the comparison of simulation results, the GA-BP model has more accurate prediction ability and better generalization ability than the traditional BP model. In one specific experiment, the model performance index decision coefficient R_2 was improved from 0.81502 to 0.95628, and the 10-experiment average prediction error was reduced from 0.0061 to 0.0041. In the future, we will work towards further improving algorithm performance. The prediction accuracy should be further improved; meanwhile, the error value should not increase. Besides, other intelligent algorithms (e.g., extreme gradient boosting) will be tested and tailored for this industrial context.

Data Availability

The data that support the findings of this study are available on request from the corresponding author.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] D. Bogdan and O. Ion, "Octane number estimation using neural networks," *Revista de Chimie*, vol. 65, pp. 599–602, 2014.

[2] Y. Ryzhov, E. S. Strizhakova, and A. Lapidus, "Modeling the octane numbers of alkenes by the inverse function method," *Petroleum Chemistry*, vol. 51, no. 5, pp. 354–362, 2011.

[3] S. Kish, A. Rashidi, H. Aghabozorg, and L. Moradi, "Increasing the octane number of gasoline using functionalized carbon nanotubes," *Applied Surface Science*, vol. 256, no. 11, pp. 3472–3477, 2010.

[4] S. Wang, S. Liu, J. Zhang, X. Che, Z. Wang, and D. Kong, "Feasibility study on prediction of gasoline octane number using NIR spectroscopy combined with manifold learning and neural network," *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 228, article 117836, 2020.

[5] B. Li and C. Qin, "Predictive analytics for octane number: a novel hybrid approach of KPCA and GS-PSO-SVR model," *IEEE Access*, vol. 9, pp. 66531–66541, 2021.

[6] X. Zhou, X. Xu, W. Liang, Z. Zeng, and Z. Yan, "Deep-learning-enhanced multitarget detection for end-edge-cloud surveillance in smart IoT," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12588–12596, 2021.

[7] F. Lu, R. Niu, Z. Zhang, L. Guo, and J. Chen, "A generative adversarial network-based fault detection approach for photovoltaic panel," *Applied Sciences*, vol. 12, no. 4, p. 1789, 2022.

[8] D. Paulo, J. Barros, and P. Barbeira, "A PLS regression model using flame spectroscopy emission for determination of octane numbers in gasoline," *Fuel*, vol. 176, pp. 216–221, 2016.

[9] X. Sun, G. Yuan, and J. M. Dai, "Multi-spectral thermometry based on GA-BP algorithm," *Spectroscopy and Spectral Analysis*, vol. 27, no. 2, pp. 213–216, 2007.

[10] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[11] M. Zou, L. Xue, H. Gai, Z. Dang, S. Wang, and P. Xu, "Identification of the shear parameters for lunar regolith based on a GA-BP neural network," *Journal of Terramechanics*, vol. 89, pp. 21–29, 2020.

[12] W. Liang, Y. Hu, X. Zhou, Y. Pan, and K. Wang, "Variational few-shot learning for microservice-oriented intrusion detection in distributed industrial IoT," *IEEE Transactions on Industrial Informatics*, p. 1, 2021.

[13] J. Cooper, K. Wise, J. Groves, and W. T. Welch, "Determination of octane numbers and Reid vapor pressure of commercial petroleum fuels using FT-Raman spectroscopy and partial least-squares regression analysis," *Analytical Chemistry*, vol. 67, no. 22, pp. 4096–4100, 1995.

[14] H. Chung, S. Cho, Y. Toyoda, K. Nakano, and M. Maeda, "Moment combined partial least squares (MC-PLS) as an improved quantitative calibration method: application to the analyses of petroleum and petrochemical products," *Analyst*, vol. 131, no. 5, pp. 684–691, 2006.

[15] J. Kelly, C. Barlow, T. Jinguji, and J. B. Callis, "Prediction of gasoline octane numbers from near-infrared spectral features in the range 660-1215 nm," *Analytical Chemistry*, vol. 61, no. 4, pp. 313–320, 1989.

[16] P. Flecher, W. Welch, S. Albin, and J. B. Cooper, "Determination of octane numbers and Reid vapor pressure in commercial gasoline using dispersive fiber-optic Raman spectroscopy," *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 53, no. 2, pp. 199–206, 1997.

[17] G. Fodor, K. Kohl, and R. Mason, "Analysis of gasolines by FT-IR spectroscopy," *Analytical Chemistry*, vol. 68, no. 1, pp. 23–30, 1996.

[18] N. P. Kardamakis and N. Pasadakis, "Autoregressive modeling of near-IR spectra and MLR to predict RON values of gasolines," *Fuel*, vol. 89, no. 1, pp. 158–161, 2010.

[19] K. Brudzewski, A. Kesik, K. Kołodziejczyk, U. Zborowska, and J. Ulaczyk, "Gasoline quality prediction using gas

chromatography and FTIR spectroscopy: an artificial intelligence approach," *Fuel*, vol. 85, no. 4, pp. 553–558, 2006.

[20] R. Balabin, R. Safieva, and E. Lomakina, "Comparison of linear and nonlinear calibration models based on near infrared (NIR) spectroscopy data for gasoline properties prediction," *Chemometrics and Intelligent Laboratory Systems*, vol. 88, no. 2, pp. 183–188, 2007.

[21] I. Litani-Barzilai, V. B. Sela, V. Bulatov, I. Zilberman, and I. Schechter, "On-line remote prediction of gasoline properties by combined optical methods," *Analytica Chimica Acta*, vol. 339, no. 1-2, pp. 193–199, 1997.

[22] Y. Deng, H. Xiao, J. Xu, and H. Wang, "Prediction model of PSO-BP neural network on coliform amount in special food," *Saudi journal of biological sciences*, vol. 26, no. 6, pp. 1154–1160, 2019.

[23] C. Huang, Y. Zhao, W. Yan, Q. Liu, and J. Zhou, "A new method for predicting crosstalk of random cable bundle based on BAS-BP neural network algorithm," *IEEE Access*, vol. 8, pp. 20224–20232, 2020.

[24] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan, "Generative adversarial networks," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–38, 2021.

[25] R. Zong, Y. Zhi, B. Yao, J. Gao, and A. A. Stec, "Classification and identification of soot source with principal component analysis and back-propagation neural network," *Australian Journal of Forensic Sciences*, vol. 46, no. 2, pp. 224–233, 2014.

[26] X. Zhou, X. Xu, W. Liang et al., "Intelligent small object detection for digital twin in smart manufacturing with industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1377–1386, 2022.

[27] X. Li, S. Dong, H. S. Mohamed, G. Al Aqel, and N. Pirhadi, "Prediction of tubular T/Y-joint SIF by GA-BP neural network," *KSCE Journal of Civil Engineering*, vol. 24, no. 9, pp. 2706–2715, 2020.

[28] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 968–979, 2020.

[29] X. Xue, Y. Li, X. Yang, X. Chen, and J. Xiang, "Prediction of slope stability based on GA-BP hybrid algorithm," *Neural Network World*, vol. 25, no. 2, pp. 189–202, 2015.

[30] Z. Cao, N. Guo, M. Li, K. Yu, and K. Gao, "Back propagation neutral network based signal acquisition for Brillouin distributed optical fiber sensors," *Optics Express*, vol. 27, no. 4, pp. 4549–4561, 2019.

[31] V. Kostenko and A. Frolov, "Self-learning genetic algorithm," *Journal of Computer and Systems Sciences International*, vol. 54, no. 4, pp. 525–539, 2015.

[32] C. Wei, J. Qin, L. Hao, J. Guo, and Y. X. Chen, "A noise based medical elites silence model and public health opinion distortion in social networks," *Frontiers in Public Health*, vol. 9, 2022.

[33] X. Zhou, W. Liang, W. Li, K. Yan, S. Shimizu, and K. Wang, "Hierarchical adversarial attacks against graph neural network based IoT network intrusion detection system," *IEEE Internet of Things Journal*, 2021.

[34] J. Guo, S. Chen, Y. Li, J. L. Li, and J. Lu, "A blockchain and IoT based lightweight framework for enabling information transparency in supply chain finance," *Digital Communications and Networks*, 2022.

[35] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, USA, 2019.

[36] Y. Quan and Q. Feng, "A hybrid genetic algorithm for twice continuously differentiable NLP problems," *Computers & Chemical Engineering*, vol. 34, no. 1, pp. 36–41, 2010.

[37] Y. Huang and M. Fei, "Motion planning of robot manipulator based on improved NSGA-II," *International Journal of Control, Automation and Systems*, vol. 16, no. 4, pp. 1878–1886, 2018.

[38] X. Zhou, X. Yang, J. Ma, and K. Wang, "Energy efficient smart routing based on link correlation mining for wireless edge computing in IoT," *IEEE Internet of Things Journal*, 2021.

[39] J. Zhou and Q. Ma, "Establishing a genetic algorithm-back propagation model to predict the pressure of girdles and to determine the model function," *Textile Research Journal*, vol. 90, no. 21-22, pp. 2564–2578, 2020.

[40] J. Liu, H. Wang, Y. Sun, C. Fu, and J. Guo, "Real-coded quantum-inspired genetic algorithm-based BP neural network algorithm," *Mathematical Problems in Engineering*, vol. 2015, Article ID 571295, 10 pages, 2015.

[41] C. Li, J. Li, Y. Li, L. He, X. Fu, and J. Chen, "Fabric defect detection in textile manufacturing: a survey of the state of the art," *Security and Communication Networks*, vol. 2021, Article ID 9948808, 13 pages, 2021.

[42] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[43] Y. Zeng, J. Chen, N. Jin, Y. Du, and X. Jin, "Air quality forecasting with hybrid LSTM and extended stationary wavelet transform," *Building and Environment*, vol. 213, article 108822.