Research on Product Classification and Demand Prediction Based on ALSTM-CNN

Jiaqi Li*, Zhiyuan Yu, Haimei Zhou

School of Harbin Engineering University, Harbin, China, 150001

* Corresponding Author Email: 18724681137@163.com

Abstract. The rapid development of e-commerce platforms has prompted a large number of retailers to join and store goods through logistics warehouses provided by the platform, which are then centrally managed by the platform. In this context, the rise of artificial intelligence and machine learning technology has brought revolutionary impact to the e-commerce industry. Under this framework, accurate sales forecasting has become the core link. This article aims to integrate machine learning models and time series analysis, aiming to improve the accuracy of sales data prediction. This article is based on the K-Means algorithm for unsupervised multi-level clustering, determining the optimal K value through elbow rules and contour coefficients. After labeling the classification data with class labels, objectively evaluate the clustering effect by combining the Calinski Harabasz index and visualizing SKUs after each clustering. Finally, using evaluation metric 1-wmape, the performance of ARIMA-ANN, BiLSTM, GRU, CNN-LSTM, Stack LSTM, and three tree models in predicting SKU sales was compared. We compared the accuracy of 15 day recursive prediction and 15 day one-time output, and selected sliding window sizes of 3, 5, 7, and 14. After comparison, it was determined that the BiLSTM model performed best in predicting 15 day sales volume and a 3-day sliding time window under multiple sample categories; ARIMA-ANN performs the best in recursive prediction of 15 day sales volume in a few sample categories. By adjusting the ARIMA parameters through ACF and PACF, the maximum predictive performance is ensured.

Keywords: BiLSTM Model, ARIMA-ANN, K-Means Algorithm, Feature Engineering.

1. Introduction

In the past few years, e-commerce platforms have experienced significant development, leading to an increasing number of retailers participating in this digital sales channel. These merchants usually store their goods in logistics warehouses provided by e-commerce platforms, which are responsible for centralized inventory management. With the rise and popularization of artificial intelligence and machine learning technology, intelligent decision-making technology has begun to find applications in the e-commerce industry. Especially, the construction of intelligent supply chains not only improves efficiency but also significantly improves consumer service quality and shopping experience by reducing inventory costs and optimizing logistics routes for goods. Accurate prediction is a crucial foundation in the intelligent decision-making process. Only when the predicted data is accurate enough can the formulation of strategies be more timely and effective, and significant cost savings be achieved.

2. Research on Establishing Model Prediction Accuracy

2.1. Time series structure transformation

After completing the preliminary data preprocessing steps, this study performed further structural transformation operations on the SKU encoded dataset. Convert the dataset from Long Format to Wide Format. In long format data, each row is an observation record, including fields such as date, product demand (qty), SKU code (sku), primary classification (category1), and secondary classification (category2). The converted wide format data reorganizes these fields so that each SKU code corresponds to a row in the data table, and the column identifier is expanded to include the actual

date field and classification features related to each SKU. This transformation effectively displays the actual features of each SKU, facilitating subsequent clustering analysis and other operations.

The accuracy indicators for model prediction established in this article are as follows:

$$1-wmap=1-\frac{\sum|y_i-\widehat{y}_i|}{\sum y_i} \tag{1}$$

Where y_i is the actual demand for the i-th sequence (the daily quantity of various goods stored by merchants in each warehouse), and $\widehat{y}_i$ is the predicted demand for the i-th sequence.

2.1.1. Profile coefficient

The contour coefficient combines the similarity within the sample and its cluster, as well as the dissimilarity with the nearest other clusters, and its calculation formula is as follows:

$$S=\frac{b-a}{\max(a,b)} \tag{2}$$

Where a is the average distance between each sample and other samples in the same cluster, and b is the average distance of the cluster where the nearest sample is located. The closer the value range is to 1, the better the clustering effect.

2.1.2. Calinski Harabasz index

The Calinski Harabasz index is an unsupervised evaluation metric that is the ratio of the covariance within a cluster to the covariance between clusters. The larger the index, the better the clustering effect. Its calculation formula is as follows:

$$s=\frac{\frac{SS_B}{k-1}}{\frac{SS_W}{N-k}}=\frac{SS_B(N-k)}{SS_W(k-1)} \tag{3}$$

Where k represents the number of clustering categories, and N represents the total number of data.

The calculation methods for SS_B and SS_W are as follows:

$$\begin{aligned} SS_B&=tr(B_k)\\ B_k&=\sum_{q=1}^{k}nq(c_q-c_E)(c_q-c_E)^T \end{aligned} \tag{4}$$

$$\begin{aligned} SS_W&=tr(W_K)\\ W_k&=\sum_{q=1}^{k}\sum_{x_q}(x-c_q)(x-c_q)^T \end{aligned} \tag{5}$$

Among them, c_q is the particle of class q, c_E is the center point of all data, and n_q is the total number of class q. Trace only considers the elements on the diagonal of the matrix, that is, the Euclidean distance from all points in class q to the class.

2.1.3. Davies Bouldin Index (DBI)

The Davidson Boding Index (DBI), also known as the classification accuracy index, is an indicator used to evaluate the quality of clustering algorithms. Assuming there are m time series, cluster these time series into n clusters. Set m time series as input matrix X and n cluster classes as N as parameters to be passed into the algorithm. Calculate using the following formula:

$$DBI = \frac{1}{N}\sum_{i=1}^{N}\max_{j\neq i}\left(\frac{\bar{s}_i + \bar{s}_j}{\left\|w_i - wj\right\|_2}\right) \tag{6}$$

3. Extraction of temporal features based on ALSTM-CNN

This article first utilized time series analysis libraries such as tsfresh and tsfel for feature extraction. Although these tools are quite efficient in feature extraction, by printing these time-series features, we found that for most SKU time-series data, due to the presence of a large number of zero values, the extracted features have a large number of missing values. The sparsity of these features cannot meet the subsequent prediction or classification needs of the model well, and manual feature selection is required, which is very cumbersome.

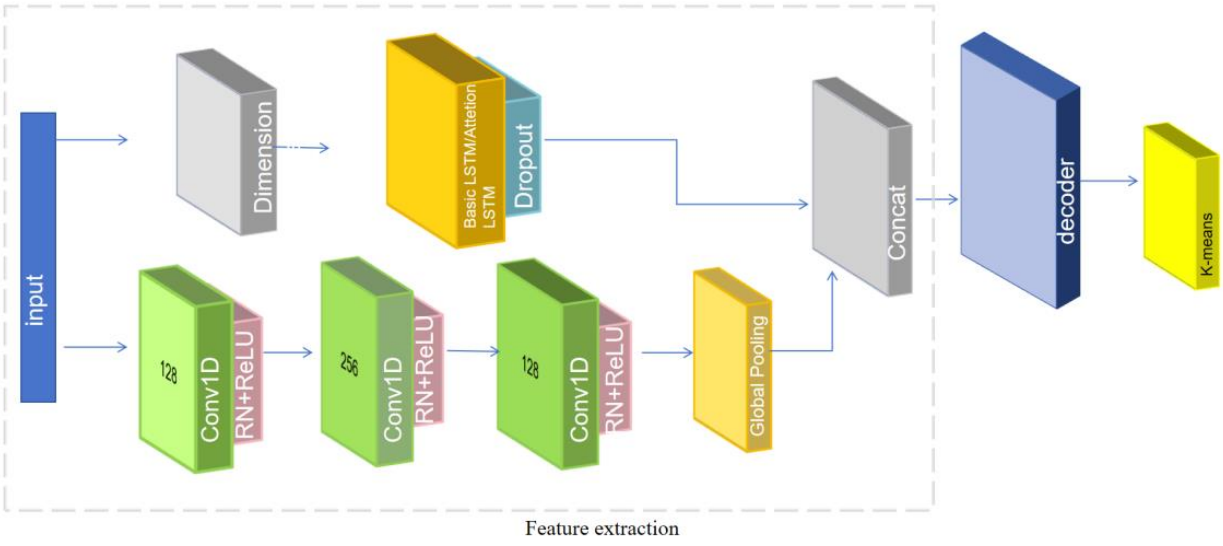

Figure 1. ALSTM-CNN schematic diagram

In order to avoid the drawbacks of manual feature selection and capture the short-term local features and long-term dependencies of time series, the feature extraction process was improved, and a deep learning strategy was chosen, especially the method combining convolutional neural networks (CNN) and long short-term memory networks (LSTM)[1]. An autoencoder architecture was introduced, as shown in Figure 1. This architecture performs automatic feature engineering in continuous convolutional layers, while transforming univariate time series into multivariate inputs through dimension shuffling. The LSTM layer captures long-term dependencies in the time series and enhances the model's ability to focus on key information through attention mechanisms. Through this architecture, the paper extract features from the time series after data cleaning. In terms of specific model architecture, the paper defined an input-output layer, followed by three convolutional layers, each with batch normalization and activation functions. On the other hand, the paper shuffled the feature dimensions and utilized the advantage of LSTM's ability to read multivariate time series features. The paper transformed univariate time features into multivariate time features and added attention mechanisms to strengthen key features. The paper added a Dropout layer and integrated the features of the convolutional layer with LSTM through global average pooling and global maximum pooling. Then, the decoding part of the autoencoder used a fully connected layer and a reconstruction layer to reconstruct the input data, matching the features with each SKU for unsupervised learning classification in the future. For prediction, the model is trained using the Adam optimizer with mean square error as the loss function.

3.1. SKU Time Series Classification Feature Construction

This article visualizes and analyzes the classification characteristics of each SKU, including the first level classification, second level classification, warehouse classification, and merchant size[2]. As shown in Figure 2, it is found that the demand for SKUs shows a certain correlation with the classification characteristics such as the warehouse type and merchant size to which they belong. In order to consider these key factors during the model classification stage, the one hot encoding method was used to transform the classified features of SKUs, and these encoded features were combined with features extracted from time series to construct a more comprehensive prediction model. Due to the dependency of time series data, the effectiveness of k-means clustering is affected. Therefore, the construction of classification features does not include qty. This feature fusion strategy aims to improve the accuracy and interpretability of the model for predicting SKU demand.

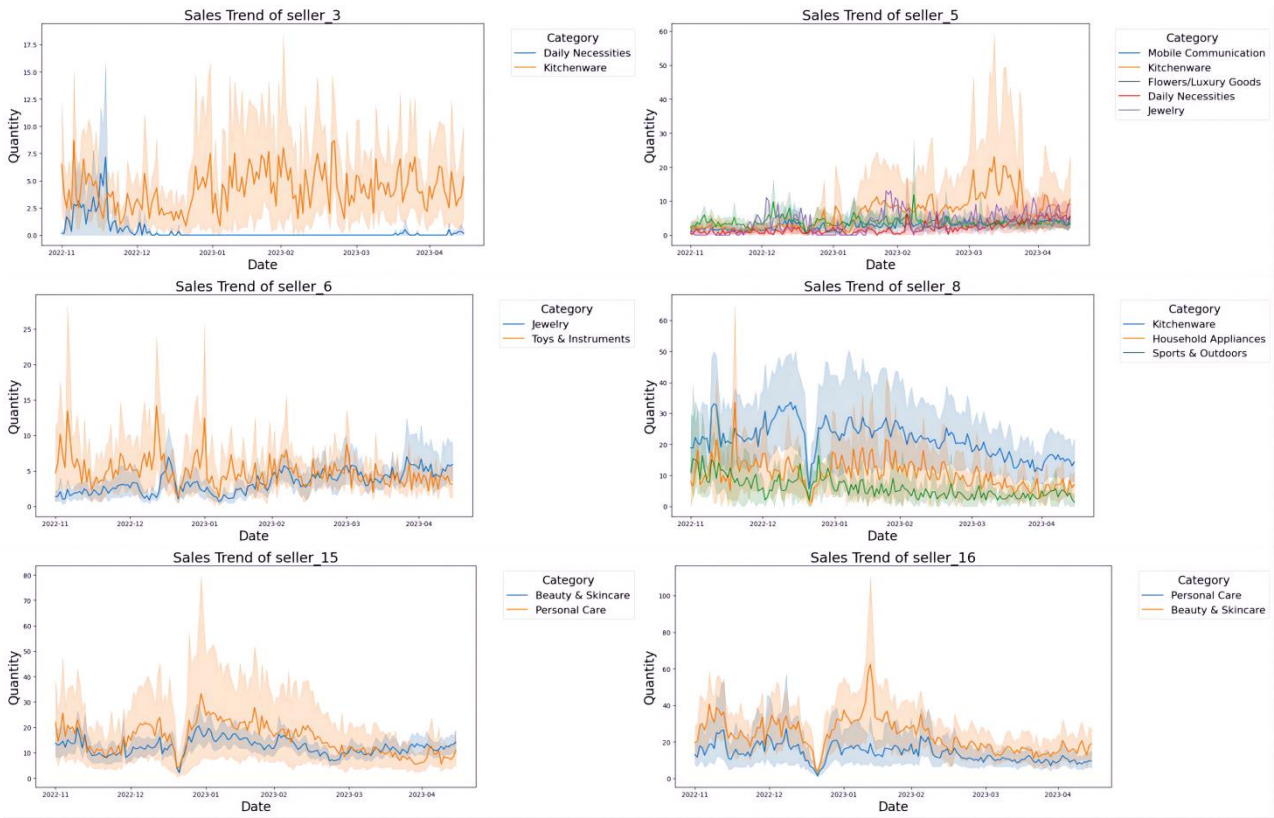

Figure 2. Visualization of primary classification, secondary classification, and warehouse classification of goods

3.2. SKU Time Series Prediction Feature Construction

Associate these extracted time features with historical demand data to form a comprehensive feature set, which is integrated as the basis for training prediction models. The aim is to predict the demand for goods at specific future time points or time periods, and based on this, select different sliding time window times to explore the impact on prediction accuracy.

3.3. Unsupervised classification based on k-means

The K-Means algorithm has been applied to features previously extracted by autoencoders combining CNN and LSTM layers, as well as features encoded by unique heat codes, which capture the essential characteristics of time series data[3]. By utilizing random seeds to ensure the reproducibility of the results, this paper calculated the clustering labels for each SKU and merged these labels with the corresponding SKUs to form a new data framework.

When conducting cluster analysis, it is necessary to determine the optimal number of clusters. To address this issue, this article employs two techniques, elbow method and silhouette score, to evaluate

the clustering performance under different numbers of clusters. This article uses the KMeans clustering algorithm to iteratively calculate different numbers of clusters (k values ranging from 2 to 5). For each k value, initialize the K-Means algorithm and fit the encoding features previously obtained through the autoencoder structure[4]. After fitting, record the inertia of each model, which is the sum of squared distances from the data point to its nearest cluster center. In the elbow method, the rate of inertia reduction slows down as the value of k increases, forming an "elbow" at a certain point, which is usually considered a good indicator of the optimal number of clusters. At the same time, the contour coefficients for each k-value were calculated, which measure the similarity of samples in the same cluster and the dissimilarity of samples in different clusters. The range of contour coefficient values is from -1 to 1, with higher values indicating that objects within the cluster are relatively dense and objects between clusters are relatively separated, making it an indicator of good clustering performance.

In this study, an iterative clustering strategy was adopted to ensure the effectiveness of the clustering results. The initial clustering analysis is completed by comprehensively considering the inertia value and contour coefficient, which together provide objective basis for determining the number and quality of clusters. However, the preliminary clustering results showed uneven distribution of categories, which may have a negative impact on subsequent data analysis.

On the premise of ensuring that the clustering quality evaluation indicators (such as contour coefficients) are within an acceptable range, this article re clusters the categories with a large number of SKUs in the initial clustering. This process was iterated three times, each iteration based on refining the previous clustering results. Through this hierarchical clustering method, large categories can be further subdivided to achieve a more detailed and balanced clustering structure.

3.4. Analysis of the validity of clustering results

By using the K-Means clustering algorithm and iteratively calculating different cluster numbers (i.e. k values) from 2 to 5. And calculate and analyze the evaluation indicators of each clustering result to ensure that our clustering quality evaluation indicators have objectivity and reliability as shown in Figure 3 and Table 1.

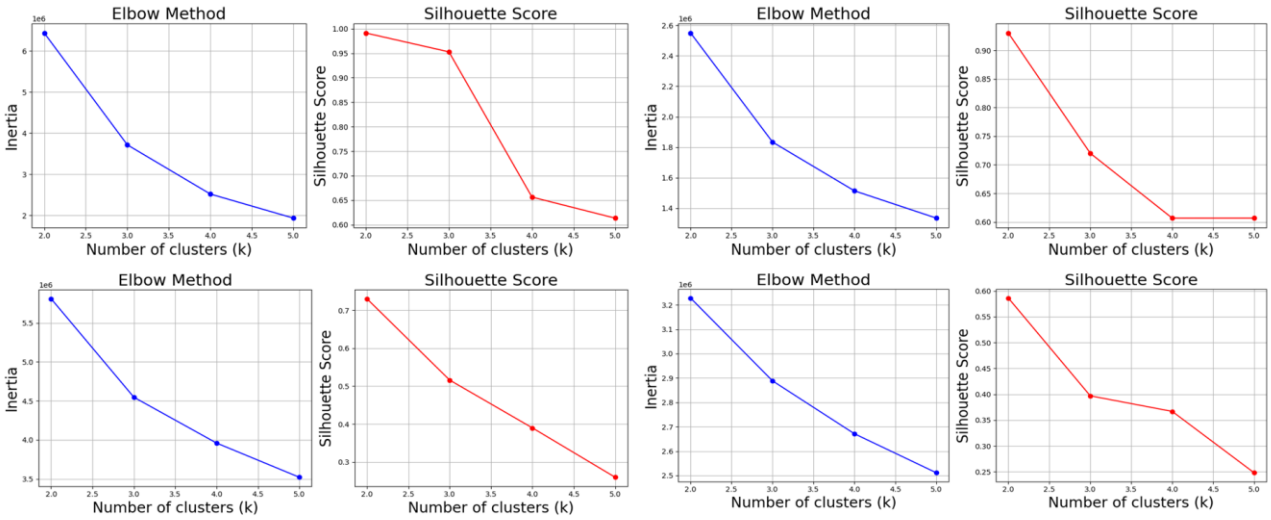

Figure 3. Elbow method for finding cluster classes

Table 1. Classification Evaluation Indicators

Clustering frequency	Davies Bouldin index	Calinski-Harabasz index	Contour coefficient
1	0.005	11299.594	0.992
2	0.572	1602.066	0.937
3	1.019	891.845	0.744
4	1.314	842.369	0.593

In order to gain a deeper understanding of the grouping performance of clustering algorithms, visualization methods were adopted to evaluate the accuracy of each classification. As shown in the figure below, even when constructing classification time features without considering product sales data, the clustering algorithm can effectively identify and separate category 5 in the first round of classification (represented by green lines in the figure). Through visual inspection, the paper observed that the time series within category 5 exhibited significant trend similarity. At the same time, when observing the other categories identified by clustering, the paper also found that there was a certain degree of trend similarity within them. This indicates that the above autoencoder can capture the intrinsic patterns of time series data. In summary, if want to classify the time series formed by these merchants, warehouses, and products, we must be able to capture the similarity of time series between different SKUs. In this paper, the constructed ALSTM-CNN is used to extract features, while also taking into account the influence of classification factors such as merchants and warehouses. When constructing time series classification features, this paper also takes into account the unique hot encoding of classification features as shown in Figure 4.

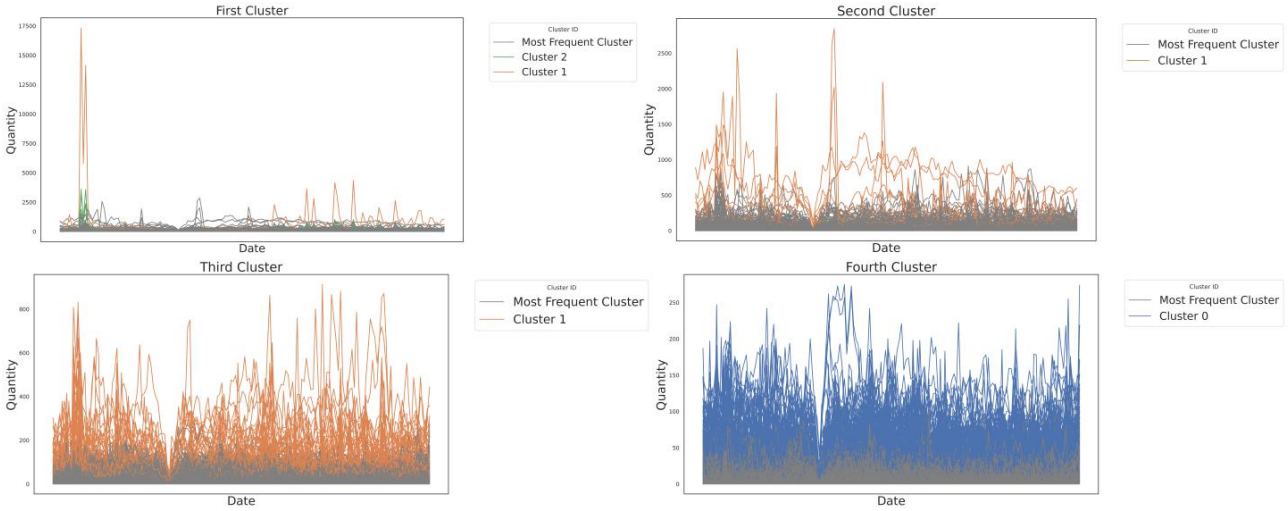

Figure 4. Visualization analysis of clustering categories

3.5. ARIMA-ANN model

The ARIMA model is one of the most important and widely used time series models, consisting mainly of three parts: autoregressive model (AR), differential process (I), and moving average model (MA). The expression formula is as follows:

$$Y_t = c + \varphi_1 Y_{t-1} + \varphi_2 Y_{t-2} + ... + \varphi_p Y_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q} + \varepsilon_t \quad (7)$$

Among them, Y_t is the time series data, $\varphi_i (i = 1,..., p)$ is the parameter of the AR model, which is used to describe the relationship between the current value and the past P time points, and $\theta_j (j = 1,..., q)$ is the parameter of MA, which is used to describe the relationship between the error of the current value and the past q time points.

Before the modeling process of ARIMA, it is necessary to ensure that the time series is stationary and non white noise data. The stationarity needs to be processed using difference, and the required difference order, d, can be selected according to the actual situation. p, q, and d are the parameters required for ARIMA. For these parameters, grid search is generally selected for automatic optimization[5]. To reduce the number of searches, the preliminary judgment of order can be made based on ACF and PACF.

But they have limitations in the linear form of the model's preconceived assumptions. Assuming that there is a linear correlation structure between time series values, the ARIMA model cannot capture nonlinear patterns, and therefore the approximate results for complex real-world problems

are not good enough. This article will use an ANN neural network model combined with it to enable the model to explain non-linear structures in time series.

Assuming that the time series consists of linear autocorrelation structures and nonlinear components, a model is established as follows:

$$y_t = L_t + N_t \tag{8}$$

Among them, L_t is the linear component and N_t is the nonlinear component. The linear component is modeled using ARIMA, and its residual is then modeled using ANN to obtain the nonlinear relationship. Then, the two predictions are combined, and the calculation process is as follows:

$$\begin{aligned} e_t &= y_t - \hat{L}_t \\ e_t &= f\left(e_{t-1}, e_{t-2}, ..., e_{t-n}\right) + \varepsilon_t \\ \hat{y}_t &= \hat{L}_t + \hat{N}_t \end{aligned} \tag{9}$$

This hybrid model utilizes the characteristics and advantages of ARIMA model and artificial neural network in determining different patterns, which may lead to better prediction performance.

3.6. Bi LSTM neural network

The Long Short Term Memory Network (Bi LSTM) is an improved LSTM method used for modeling time series data. The core idea is to simultaneously utilize the contextual information of time series data to enhance the model's understanding of the data[6]. On the basis of standard LSTM, Bi LSTM introduces a reverse LSTM layer, so that the network can learn information from two time directions. The design of Bi LSTM is particularly suitable for application scenarios that rely not only on historical information but also on future contexts. When processing natural language text or audio signals, Bi LSTM can more accurately model the characteristics of each element in a sequence by combining information from two directions. The core principle of backpropagation is as follows:

$$\begin{aligned} \overleftarrow{f}_t &= \sigma(W_{\overleftarrow{f}}[h_{t+1}, x_t] + b_{\overleftarrow{f}}) \\ \overleftarrow{i}_t &= \sigma(W_{\overleftarrow{i}}[h_{t+1}, x_t] + b_{\overleftarrow{i}}) \\ \overleftarrow{\tilde{C}}_t &= \tanh(W_{\overleftarrow{c}}[h_{t+1}, x_t] + b_{\overleftarrow{c}}) \\ \overleftarrow{C}_t &= \overleftarrow{f}_t \cdot C_{t+1} + i \cdot \overleftarrow{\tilde{C}}_t \\ \overleftarrow{o}_t &= \sigma(W_{\overleftarrow{o}}[h_{t+1}, x_t] + b_{\overleftarrow{o}}) \\ \overleftarrow{h}_t &= \overleftarrow{o}_t \cdot \tanh(\overleftarrow{C}t) \end{aligned} \tag{10}$$

In the above formula, $\overleftarrow{f}_t$ represents the reverse forgetting gate at time step t, which is used to control the model's ability to retain the previous time step unit state ct. It is the reverse input gate at time step t, which determines how much new information in the current input X_t will be updated to the unit state. $\overleftarrow{C}_{t+1}$ is a candidate value for new information, generated through the tanh activation function. $\overleftarrow{C}_t$ represents the unit state of the reverse layer at time step t, integrating information from the forget gate and input gate. $\overleftarrow{O}_t$ is the output gate that determines how much information about the unit state will be transmitted to the hidden state[7]. Finally, $\overleftarrow{O}_t$ represents the hidden state of the model's output at the time step, integrating information from the output gate and the current unit state processed by the tanh function.

3.7. CNN-LSTM

Similar to the partial functions of the extraction autoencoder constructed in the previous text, in this hybrid neural network, CNN is responsible for extracting partial features of each input time feature sequence through convolutional and pooling layers, and then passing them to LSTM to capture long-term temporal dependencies[8]. At time step t, input data, CNN extracts features and unfolds them into a one-dimensional vector V_t. Then, LSTM performs the following operations at time t:

$$\begin{aligned} V_t &= CNN(X_t) \\ h_t C_t &= LSTM(V_t, H_{t-1} C_{t-1}) \end{aligned} \tag{11}$$

A one-dimensional time series feature is input into LSTM along with the hidden state and cell state of the previous time step to generate two states for the current time step.

3.8. Stack-LSTM

Stacked LSTM is a deep learning model consisting of multiple LSTM layers stacked together in order. The core of this algorithm is that the output of one LSTM layer is not only used as input for the next time step, but also fed into another LSTM layer as its input data[9]. This design enables the network to capture more complex patterns and long-term dependencies in data, as each layer can learn different levels of sequence abstraction. The formula for calculating the model in one time step is as follows:

$$\begin{aligned} &h_t^{(1)}, C_t^{(1)} = LSTM^{(1)}(X_t, h_{t-1}^{(1)}, C_{i-1}^{(1)}) \\ &h_t^{(2)}, C_t^{(2)} = LSTM^{(2)}(X_t, h_{t-1}^{(2)}, C_{i-1}^{(2)}) \\ &\vdots \\ &h_t^{(N)}, C_t^{(N)} = LSTM^{(N)}(X_t, h_{t-1}^{(N)}, C_{i-1}^{(N)}) \end{aligned} \tag{12}$$

In the above formula, $h_t^{(1)}$ and $C_t^{(1)}$ represent the hidden state and cell state of the i-th LSTM layer at time step t, respectively. X is the input data at time step t. The state of the previous hidden layer is input, and its own hidden state is calculated until the last LSTM layer. This structure allows each layer to capture information in the data at different time scales.

3.9. XGBoost

XGBoost is a common Boost model due to the application of second-order Taylor expansion, which has higher accuracy than GBDT and can customize loss functions[10]. Assuming that K trees have been trained, the final predicted value for the i-th sample is equal to:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{13}$$

Among them, represents the sample features, represents the prediction result of the k-th tree on the sample, and these values are added together to obtain the final result. Combined with the actual results, a loss function can be constructed:

$$Obj = \sum_{n=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{14}$$

Among them, l represents the loss function, and the latter term is used to control complexity and prevent overfitting.

3.10. LightGBM

LightGBM is a decision tree algorithm based on Histogram, which has faster training speed, higher efficiency, lower memory usage, and higher accuracy. Although it is an improvement based on the GBDT algorithm, compared to GBDT, XGBoost algorithm, LightGBM, it performs well in processing massive data.[11]

Catboost has the advantages of not having to perform feature engineering on categorical features and having an adaptive learning rate for predicting offsets.[12]The calculation method for its adaptive learning rate is as follows.

$$\eta_t = \frac{1}{\sqrt{t+1}}$$
$$\alpha_t = \frac{\sum_{i=1}^{t} \eta_i}{t} \tag{15}$$

Among them, t is the number of iterations, η_t is the learning rate of the t-th iteration, and α_t is the average learning rate of the previous t-th iteration. The objective function is the squared loss function:

$$L(\theta) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{16}$$

This article expects the model to achieve good prediction results for products with high sales volume. Therefore, the product with the highest sales volume in each category after classification is selected and predicted and verified through multiple models. The calculated 1-wmap values are filled in the following Table 2

Table 2. Model Selection Table Based on 1-wmap

Category model	A	B	C	D	E	F
XGBoost	0.7981	0.8616	0.5082	0.8920	0.2142	0.0925
LightGBM	0.8172	0.8577	0.6161	0.9112	0.2204	0.0916
Catboost	0.8063	0.8499	0.5623	0.9181	0.2941	0.2296
BILSTM	0.6357	0.8431	0.8440	0.9343	0.5643	0.5539
Stacked LSTM	0.6482	0.8418	0.8819	0.9298	0.4711	0.5492
GRU LSTM	0.5699	0.8448	0.9036	0.9268	0.5456	0.5496
CNN LSTM	0.5135	0.8776	0.6700	0.9075	0.5174	0.3116
ARIMA—ANN	0.4977	0.4894	0.5024	0.8662	0.6256	0.5923

It was found that the BILSTM model has greater stability and accuracy in predicting products with higher sales in various categories. So the paper chose BILSTM for prediction and solution.

After determining the prediction model, different categories of products were randomly selected, and the prediction was adjusted by adjusting the sliding window parameters of the BILSTM model. The appropriate number of sliding windows was determined by the change in prediction accuracy. Partial results are shown in Table 3

Table 3. Time sliding window selection table for different categories of products

Number of windows	A	B	C	D	E	F
3	0.6145	0.8185	0.8440	0.9343	0.5267	0.5668
5	0.6357	0.6867	0.9157	0.8554	0.5128	0.5232
7	0.6006	0.6845	0.8764	0.9411	0.4991	0.3921
14	0.6070	0.6830	0.9001	0.9160	0.5325	0.2336

Selecting some important time nodes as parameters for the sliding window, it was found through prediction that the setting of three time sliding windows had better results.

4. Conclusions

From the calculation results of the prediction accuracy index, 1-wmap, it can be discovered that this self-encoder based on the ALSTM-CNN hybrid neural network is established for extracting SKU time series features. The results are achieved through unsupervised multi-level clustering by the K-Means algorithm, where the optimal K value is determined by the elbow rule and contour coefficient. It has an excellent effect on all kinds of prediction models. Among them, under this method of time characteristic value extraction and classification for the inventory of different commodity categories, the BiLSTM model performs better in the sliding time window for one-time prediction of the 15-day sales volume and 3-day sales volume in various categories, while ARMI-ANN exhibits a better performance in the recursive prediction of the 15-day sales volume within the category of small samples. Compared with the single time series forecasting model and the traditional time series feature extraction of commodity inventory, this approach can offer better assistance and serve as a decision-making basis for inventory management.

References

[1] Wang Ling, Zhou Nan, Shen Peng. Time series anomaly pattern recognition based on adaptive k-nearest neighbors [J]. Computer Research and Development, 2023, 60 (01): 125-139

[2] Zhou Wenshuo. Time series classification based on multi-scale fully convolutional networks and cross clustering algorithms [D]. Donghua University, 2020.

[3] Ye Yuqi Integrated decision-making of data-driven e-commerce demand prediction and inventory optimization [D]. Beijing Jiaotong University, 2022

[4] Qiu Pingping Commodity demand prediction based on sales records in the context of supply chain [D]. South China University of Technology, 2021

[5] Ying Zhou Dan Research on Sales Forecast of E-commerce Platforms in the Scenario of Business Promotion [D]. Nanjing University, 2022.

[6] Mou Shucheng Algorithm research on time series prediction and analysis for the retail industry [D]. Beijing University of Posts and Telecommunications, 2019.

[7] Teng Yue A prediction method for promotional sales of B2C e-commerce platforms [D]. Dalian University of Technology, 2022.

[8] Luo Liang Research on Green Warehouse Optimization of Company C Based on Combination Demand Forecasting [D]. Southwest University of Science and Technology, 2023.

[9] Liu Wei Retail supply chain demand prediction algorithm based on meta learning and machine learning [D]. Southeast University, 2022.

[10] Qilan Huang, Qi Liu, Lijin Guo. Prediction model of effluent quality based on BiLSTM & GRU combined algorithm [A]. Southeast University and Chinese Association of Automation, 2022.

[11] Teng Yue A prediction method for promotional sales of B2C e-commerce platforms [D]. Dalian University of Technology, 2022.

[12] Luo Liang Research on Green Warehouse Optimization of Company C Based on Combination Demand Forecasting [D]. Southwest University of Science and Technology, 2023.