Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

Research on AIGC-based Supply Chain Inventory Demand Forecasting Model and Nonlinear Optimization Strategy

Zhe Wang[1], Hongsong Xue[2,✉], Junhua Hu[1]

[1] *Wuchang Institute of Technology, Wuhan, Hubei, 430065, China*
[2] *Wuhan Qingchuan College, Wuhan, Hubei, 430065, China*

ABSTRACT
Supply chain inventory forecasting and control is an integral part of supply chain management system, and it is a focus that industries must pay attention to in their operation and management. In this paper, the supply chain inventory demand forecasting model is constructed from the perspective of supply chain end, combined with the Transformer model in AIGC technology. The DL-Informer model is used to improve the Transformer model, realize the feature fusion of graph convolutional neural network, design and solve the feature graph adjacency matrix and complete the information fusion of each feature subgraph to improve the prediction accuracy. Aiming at the problems faced by supply chain inventory demand forecasting, the traditional algorithm with strong local optimization ability is combined with the genetic algorithm, and the hybrid genetic algorithm (HGA) is proposed to solve the nonlinear optimization problem. In the supply chain inventory forecasting practice, when the forecast length is 12, the MSE, MAE and RMSE index values of this paper's forecasting model are 0.202, 0.174 and 0.416, respectively, which have more stable long-term forecasting performance compared with other models. And in the nonlinear simulation optimization experiments, the HGA algorithm shows good convergence and outstanding optimization effect in the nonlinear problem of supply chain inventory.

Keywords: AIGC, Transformer, nonlinear optimization, inventory demand forecasting, genetic algorithm

1. Introduction

Inventory problem has been a major problem for many enterprises, how to control the inventory of each node enterprise in the supply chain from the perspective of supply chain is a major hot spot in today's supply chain research. Restricted by the production capacity, each node enterprise in the supply chain needs to order and store a large number of materials required for production before

✉ Corresponding author.
E-mail address: xuehongsongqcxy@163.com (H. Xue).
Received 20 July 2024; Revised 10 October 2024; Accepted 05 February 2025; Published Online 16 April 2025.
DOI: 10.61091/jcmcc127b-176

obtaining the actual order, and produce some products with long manufacturing cycle and complex structure in advance, and establish reserve stock to cope with the peak demand [1-3]. The amount of reserve inventory is difficult to control, often far beyond the actual demand, the high cost of warehousing, taking up a large amount of money, masking the many management problems of the enterprise, and seriously impede the high-quality development of the supply chain [4-6].

To control the inventory, we must start from a good demand forecast. Demand forecasting is the basis for activities such as production planning, and inaccurate need forecasting is usually the most important factor leading to enterprise inventory [7]. However, demand forecasting is based on historical data and the decision maker's judgment of the future, and there is always a certain bias in the forecast, and roughly the formation of forecasting bias is caused by two types of factors, namely, the internal enterprises in the supply chain and the external environment [8-11]. The information structure in the supply chain is decentralized, and the local inventory status and dynamic demand forecasting information of each member cannot be shared. Under the independent demand forecasting of the node firms, each member in the supply chain dynamically maintains its independent forecasting process and induces the forecasting results into their respective inventory replenishment strategies [12-16]. Due to the dynamic nature of the supply chain, such as time delay and demand amplification, the forecasts of each member vary greatly, which ultimately generates the “whiplash effect” [17]. In today's supply chain management environment, the enterprise's inventory problem not only affects the interests of an enterprise itself, but also affects its associated upstream and downstream enterprises [18-20]. When each enterprise in the supply chain can operate healthily and efficiently, the supply chain will have strong competitiveness, which makes the member enterprises in the supply chain realize the supply chain [21-22]. Therefore, how to forecast demand and inventory control from the perspective of supply chain is an important part of supply chain management.

Many scholars have conducted detailed research specifically on supply chain inventory demand forecasting challenges.Seyedan, M. et al. compared the performance of different classes of big data analytics algorithms in supply chain demand forecasting and specifically analyzed the situation of closed-loop supply chain demand forecasting, which provided theoretical references for supply chain management [23]. Zohra Benhamida, F. et al. developed an intelligent inventory management system and proposed two methods for product demand forecasting based on the results of analyzing the historical sales and demand data of an enterprise, which provides accurate forecasting accuracy for making production and supply plans and optimizing supply chain decisions [24]. Kilimci, Z. H. et al. proposed an integration strategy for supply chain inventory demand model and introduced boosting integration strategy into the forecasting model to construct a forecasting system containing support vector regression, deep learning model and integration strategy, which significantly improves the accuracy of inventory forecasting [25]. Feizabadi, J. et al. designed a hybrid demand forecasting method based on ARIMAX and neural networks, which can effectively solve the problem of amplified differences in cuju forecasts due to distorted demand information in a multilevel supply chain, and thus improve the forecasting performance of inventory demand [26]. Deng, C. et al. developed a mathematical model of logistics and transportation for cost minimization as well as profit maximization and solved it using deep learning long and short-term memory theory for deep inventory management, which is capable of quickly detecting abnormal inventory behaviors and reducing costs over the supply chain lifecycle [27].

In this paper, a supply chain inventory demand forecasting model is constructed based on the AIGC technology and the Transformer model, which is an important technology in the AIGC technology, is selected as the method. Informer is proposed to improve the structure of traditional Transformer model, and graph convolutional neural network and residual connection are used to further optimize the structure of Informer network, which is named as DL-Informer.A method is designed for solving the adjacency matrix of feature graphs, and after obtaining the adjacency matrix, the processing of the graph structural data is completed by graph convolutional neural network. Determine the model

expression of the graph convolutional neural network, and after converting the feature map into graph data, integrate the information of each feature sub-map, and then extract the more valuable features between the data, and finally improve the prediction accuracy of the Informer. Facing the nonlinear optimization problem faced in supply chain inventory demand forecasting, hybrid genetic algorithm (HGA) is proposed as a method to solve the nonlinear optimization problem. In the two types of problems, unconstrained optimization and constrained optimization, SHGA based on the most rapid descent method and PHGA based on the penalty function method are used to solve the problems, respectively. Finally, the supply chain inventory data of an FMCG enterprise is used to carry out the application practice of supply chain inventory demand forecasting to test the practical utility of the supply chain inventory demand forecasting model in this paper, and the advantages of the HGA algorithm in solving the nonlinear optimization problems are verified through the simulation cases and examples.

2. AIGC-based supply chain inventory demand forecasting model

Today is an era of rapid development of economy and information technology, the competition among market participants is becoming increasingly fierce, especially the mutual competition among enterprises is gradually shifting towards the competition among supply chains, which has led to the widespread dissemination of the idea of supply chain management, in which the most important inventory forecasting and control has become a focus of attention for various industries in the operation and management, and therefore the research on inventory forecasting and control from the end of the supply chain has an important significance. Therefore, the research on inventory forecasting and control from the end of supply chain has important significance. Along with AIGC technology being sought after by various industries, it has become an inevitable trend to be applied to the research of supply chain inventory forecasting. In this paper, we will combine the AIGC technology with the transformer model, i.e., Transformer model, to construct a supply chain inventory demand forecasting model, aiming at providing theoretical support for enterprise inventory forecasting and control, and further enriching and developing the theory and method of enterprise inventory control in the supply chain environment.

2.1. AIGC technology

Artificial Intelligence Generated Content (AIGC) represents a major leap forward in the digital economy, addressing the challenges of digital intelligence by automating the generation of a variety of content [28]. This technology has made significant progress since the development of the underlying models in the 1950s, and AIGC has grown even more dramatically in recent years due to deep learning and the adoption of various architectures.

AIGC has evolved from simple automated content generation to providing creative and personalized content solutions. AIGC relies on three key components: data, hardware, and algorithms, which not only change the way content is produced, but also reshape the user experience and interaction patterns. High-quality data is essential for training accurate models, while powerful computing hardware is needed to process large data sets and run complex algorithms. The algorithms themselves, especially those based on various architectures, determine the quality of the content generated. The interplay between data, hardware, and algorithms is critical to the success of AIGC, enabling it to generate high-quality, diverse text, image, audio, and video content.

The cornerstone of today's robust AIGC world consists of several important techniques, which are Generative Adversarial Networks (GANs), Variational Auto-Encoders (VAEs), Regression Models, CLIP Models, Transformer Models, Diffusion Models, and so on. This study focuses on the transformer model, which is also known as the Transformer model.

2.2. Transformer model

The Transformer model has an encoder-decoder structure, which is a loop-avoiding network architecture where word sequences are first input to the encoder and then output from the encoder to the decoder. Since the inputs and outputs of the encoder and decoder are sequences of tokens with the same dimensions, the encoder and decoder can be stacked together to obtain high-level semantic features by deepening the number of network layers [29]. However, as the number of network layers deepens, the distribution of data features will always change. Transformer ensures a stable distribution of data features by introducing a layer normalization operation, which reduces the loss of information and makes the training of the network smoother. In the following, we will introduce the important components of the Transformer model structure one by one.

1) Position encoding. Because the Transformer model does not have the structure of a recurrent layer, it does not have the ability to learn word order information like recurrent neural networks, so it introduces positional coding to utilize the word order information specific to the input word sequences.The original input to the Transformer model is word vectors without word order information, and the positional coding combines the absolute or relative positional information with the word vectors to form a new representation. Position encoding combines absolute or relative positional information with word vectors to form a new representation that is input to the model so that the Transformer model can learn word order information. Positional encoding records the correlation between sequential word positions, and compared to the sequential input of recurrent neural networks, the Transformer model is able to store positional relationships between words while inputting the data in parallel, which both largely increases computational speed and reduces storage space.

In the Transformer model, the positional information is represented alternately with sine and cosine functions, which are used to generate positional encoding with different frequencies, as shown in Equation (1) and Equation (2):

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d}}\right) \tag{1}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d}}\right) \tag{2}$$

where PE is the position encoding matrix, pos represents the position of the word in the whole word sequence, d represents the total number of feature embedding dimensions in the hidden layer of the Transformer model, i represents the dimensionality, $2i$ represents the even numbered dimensions, and $2i+1$ represents the odd numbered dimensions, i.e., $2i+1 \le d$, (pos, i) represents the pos th row and i th column in the position encoding matrix, which is computed by using a sine function for the even numbered columns, and by using a cosine function for the odd numbered columns. For any deviation n, PE_{pos+n} can be expressed as a linear representation of PE_{pos}, which helps the Transformer model to learn from relative positions.

2) Encoder-Decoder. Transformer's encoder contains a multi-head self-attention layer and a fully-connected by-position layer in each layer.Transformer's decoder also consists of a stack of six identical decoder sublayers. Each layer of the decoder's structure contains a masking multi-head self-attention layer, a multi-head self-attention layer, and a fully-connected layer processed by position.The Transformer model is processed using residual joins and layer normalization (LN) operations after each encoder and decoder sub-layer.The Transformer is made up of multiple sub-layers stacked on top of one another, and the use of layer normalization in each of these sub-layers can ensure a stable sequence of features.

3) Self-attention mechanism. The self-attention mechanism of Transformer model can capture the intrinsic relevance information of features or data without relying on extrinsic information. In natural language processing tasks, Transformer relies on the self-attention mechanism to compute the attention between each word, so that each word has global semantic information, thus capturing the global context of the entire sequence of words with long-range dependencies. Prior to the advent of Transformer, the two more prevalent methods of attention computation were dot-product and additive attention.Transformer uses scaled dot-product attention, which utilizes dot-products to compute similarity. Scaled dot product attention is able to accommodate higher input dimensions, leading to faster computation than other attention computation methods.

After linearly transforming the input data into query Q, key K and value V, the scaled dot product attention first obtains the dot product of each query Q and each key K, uses this result to divide by the square root of the dimensions of query Q and key K, and then processes it through the softmax operation to obtain the self-attention score matrix, treats the self-attention score matrix as a weight for value V, and utilizes the value V and the self-attention score matrix to compute the self-attention score matrix. The final output result matrix of the attention layer is calculated using value 9 and the self-attention score matrix. Equation (3) describes how the self-attention is calculated:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3}$$

where $\sqrt{d_k}$ represents the square root of the dimension of key K and $softmax(QK^T/\sqrt{d_k})$ represents the matrix of self-attention scores, where the scaled factor (scaled factor) can be seen as $1/\sqrt{d_k}$.

4) Multiple self-attention. Multiple heads in multi-head self-attention focus on unlike global and local information, which endows the Transformer model with great power as it is able to extract richer and more comprehensive features. Compared to single-head self-attention, multi-head self-attention uses different linear projections to project query Q, key K and value V to different spaces, and computes the self-attention for each eye in different projection spaces, so as to obtain features from all aspects of the input and get better experimental results. Specifically, the Transformer model projects query Q, key K, and value V multiple times, then computes scaled click attention independently using different projections, and finally splices the single-head self-attention computations of all heads to obtain multi-head self-attention. The multi-head self-attention is formed by combining multiple self-attention as shown in Eq. (4) and Eq. (5):

$$MHSA(Q,K,V) = concat(head_0,...,head_h)W^0 \tag{4}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{5}$$

where W_i^Q ð $^{d\times d_k}$, W^0 ð $^{hd_0\times d}$, h represent the number of heads in the multi-head self-attention, W_i^Q represents a learnable linear transformation matrix that maps query Q into the ith projection space, and W_i^K and W_i^V are the same. W^O is also a Transformer model learnable linear transformation matrix whose function is to map the self-attention obtained from splicing back to the original dimension of the input token after splicing all the heads.

2.3. Transformer-based supply chain inventory demand forecasting model

Transformer can efficiently extract features and relationships between data in parallel due to the use of positional coding and multi-head self-attention mechanism to process the input data, but for a large amount of data, Transformer also increases the parameters of each matrix vector in the computation of the attention mechanism in the square level, which greatly reduces the computational efficiency,

and also improves the computational complexity and memory This greatly reduces the computational efficiency and increases the computational complexity and memory consumption. Therefore, many researchers are committed to improving the structure of Transformer to improve efficiency, and Informer is one of the improved models [30].

2.3.1. DL-Informer Improvement Model Principles. In the Informer model, in order to improve the robustness of the model, the encoder is usually designed as a stack structure, which divides the encoder into two parts, the main sequence and the replica sequence, with the replica sequence accounting for half of the length of the main sequence, and gradually reduces the number of self-attention layers by deleting one layer at a time, so as to ensure that the output dimensions are consistent with those of the main sequence. Assuming that the initial input data length is L, the length of the main sequence is L and the length of the replica sequence is L/2. After the multi-head attention mechanism and one-dimensional convolutional pooling, the length of the sequences both become 1/2 of the original. If the number of stack layers of the encoder is 3, the main sequence will undergo two pooling operations, and the replica sequences will undergo one pooling operation, and the final feature map consists of the main sequence and replica sequences with both lengths of L/4. The stack structure of the Informer encoder allows the model to learn more feature information, but for the merging of the main sequence and replica sequences at the end, the original paper has only performed a simple splicing, without considering the relationship between the feature subgraphs. And the Informer model may cause network degradation phenomenon or gradient disappearance and gradient explosion phenomenon with the increase of coding layer depth, resulting in slow convergence of the model and inaccurate prediction results.

To address the above problems, this paper improves the encoder structure of Informer and names it DL-Informer.The specific improvements are as follows: for the encoder's stack output, a simple splicing is performed first, and then graph convolutional neural network (GCN) is utilized to analyze the relationship between the feature subgraphs, and the features are fused into the feature graph output; then by introducing residual connections, the network can be transfer across layers to deeper levels, thus reducing the degradation problem of the network [31]. The number of encoder stacks in DL-Informer is 3 layers.

The core improvement of DL-Informer lies in feature fusion with graph convolutional neural networks. Graph Convolutional Neural Network is a commonly used deep learning network for graph data with the ability to automatically extract spatially localized features in the feature graph and establish relationships between these features for better processing of features. Graph convolutional neural networks mainly process graph structure data, which consists of nodes V and edges E in the graph, i.e., $G=(V,E)$. where $v_i \in V$ denotes a node in the graph and $e_i \in E$ denotes an edge in the graph. In graph structure data, nodes usually denote vectors with d-dimensional features, the number of nodes is denoted by n, and the relationship between vectors is represented by the adjacency matrix. The representation of nodes and adjacency matrix is shown in the following equation:

$$X \in R^{n \times d} \tag{6}$$

$$a_{ij} = \begin{cases} 1, & e_{ij} \in E \\ 0, & e_{ij} \notin E \end{cases} \tag{7}$$

In graph data, a graph can be constructed using individual samples as nodes and using the connection relationships between the samples. Therefore, for the feature graph output by the Informer encoder, the vector at each position in the feature graph will be regarded as a node in the graph, and the node-to-node relationship is represented by the adjacency matrix. According to the characteristics of the data output from the Informer encoder, this chapter designs a method to solve the adjacency matrix of the feature graph, solves the product between the corresponding vectors of two nodes, and

compares the result with 0. If the result is greater than 0, it indicates that the two nodes are connected, and the value corresponding to the adjacency matrix is 1. Otherwise, it indicates that the two nodes are not connected, and the value corresponding to the adjacency matrix is 0, and the adjacency matrix is denoted as shown below:

$$m_{ij} = \sum_{k=1}^{n} x_{ik} x_{kj} \tag{8}$$

$$a_{ij} = \begin{cases} 1, & m_{ij} > 0 \\ 0, & m_{ij} \le 0 \end{cases} \tag{9}$$

where m_{ij} denotes the result of multiplying two node vectors, n is the feature dimension of each node, x_{ik} denotes the value of the k th dimension of the i th node, and a_{ij} is the value of the corresponding point of the adjacency matrix.

The graph convolution neural network will perform convolution operation on each node and neighbor matrix of the input feature graph, and then change the feature representation of each node to get more diversified features, the model expression of graph convolution neural network is shown in the following equation:

$$Z = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} XW) \tag{10}$$

where $\tilde{A} = A + I_N$, A denotes the adjacency matrix, I_N is the unit matrix; $\tilde{D} = \sum_m \tilde{A}_{im}$ denotes the degree matrix; X denotes the feature vector before the graph neural network processing; z is the feature vector after the graph neural network processing; W denotes the trainable parameter matrix.

After converting the feature graph into graph data, graph neural network can then be used to analyze the relationship between the nodes in the feature graph and complete the information fusion of the feature subgraphs, thus extracting the more valuable features between the data, thus improving the prediction accuracy of Informer.

The graph convolutional neural network solves the problem of feature fusion in the output of the encoder stack, however, it cannot solve the problem of model gradient degradation. As the depth of the network increases, the information obtained by the model becomes less and less, and the model may suffer from the phenomenon of gradient disappearance or gradient explosion during training, at which time the problem can be solved by introducing residual connections in the network. The residual connection is directly jumped from the first multi-attention mechanism to the final encoder feature map, which is added with the output of the encoding layer that has gone through multiple layers as the output of the feature map. Since the pooling operation in the encoding layer reduces the dimension of the output information, the residual-connected multi-head attention mechanism needs to be dimensionally transformed with a 1×1 convolutional layer to ensure the same shape.

2.3.2. DL-Informer inventory demand forecasting methodology. DL-Informer solves the problems of Informer when it comes to feature fusion and multilevel gradient degradation, and it will be used in this chapter to forecast inventory demand.

The prediction model consists of five parts, namely the data preprocessing part, the coded embedding part, the encoder part, the decoder part and the fully connected output part. The data preprocessing part adopts the preprocessing methods in Chapter 3, i.e., error data repair, data normalization and string data encoding process, but attention should be paid to the format of the input data, which should comply with the input requirements of the DL-Informer encoder and decoder; the coded embedding part includes the positional coding, token coding and temporal coding, which are used to extract more features from the original data, and to map the feature dimension mapping to the high dimensional space, which in turn facilitates the feature processing of the encoder and decoder;

the encoder part is designed with three layers, two of which are composed of sparse self-attention and distillation layers, and the last layer is a sparse self-attention layer; the encoder stack is designed as 3, 2, 1, i.e., it means that there are one main sequence and two copy sequences, which need to be processed by the graphical convolution neural network for the feature fusion process, in which the main sequence is also added with residual connections to avoid network degradation; the decoder part is designed with 2 layers, each layer consists of a masked multi-attention layer and a multi-attention layer; the fully-connected output part will output the final prediction results, and then iteratively seek the optimal parameters of the model through back propagation to complete the training operation of the model.

2.4. Experimental analysis

In order to verify the superiority of the supply chain demand forecasting model proposed in this paper on inventory demand forecasting, Google cluster data will be used as the research data in this section, and LSTM, BiLSTM, Temporal Convolutional Networks (TCNs), Gated Recurrent Units (GRUs), and Informer model will be selected as the comparison models to carry out performance comparison experiments by using the data related to load sequences, the amount of CPU resource usage and RAM resource usage in executing the tasks in Google cluster data, respectively. The performance comparison experiments are conducted using load sequences from Google cluster data, CPU resource usage and RAM resource usage, respectively.

The results of the experiments using load sequences from Google cluster data are shown in Table 1. Among the many comparative models, the MSE value of this paper's model is the lowest among all models, which is only 91.34, and the R^2 index value is 42.91% higher than that of the LSTM model with the lowest R^2 index value. The prediction time and training time are 1.10973s and 3.34986s respectively, which are only slightly inferior to the GRU model.

Table 1. Performance comparison of models in load sequence of Google cluster data

Model	MSE	R^2	Prediction time(s)	Training time(s)
LSTM	16749.92	0.6946	1.91871	4.81229
BiLSTM	18653.62	0.76659	3.7167	8.13328
TCN	12409.57	0.70674	6.56471	13.24954
GRU	10851.91	0.74617	0.97179	3.28062
Transformer	129.32	0.98543	1.28518	4.48859
Model of this article	91.34	0.99265	1.10973	3.34986

Performance comparison experiments are conducted on CPU usage data from Google cluster data, and the specific results are shown in Table 2. It can be seen that in the three evaluation indexes of MSE, R^2 and prediction time, the performance of this paper's model is optimal, reaching 6.68, 0.98508, and 1.15908 s. In the training time, it reaches only less than 0.1 s more than the fastest GRU model.

Table 2. Performance comparison of models in CPU usage of Google cluster data

Model	MSE	R^2	Prediction time(s)	Training time(s)
LSTM	314.06	0.81216	1.82238	4.9474
BiLSTM	331.08	0.79285	4.69131	9.58315
TCN	308.46	0.75887	6.9781	14.12311
GRU	263.8	0.83299	1.17174	3.48312
Transformer	42.56	0.97545	1.35937	4.59606
Model of this article	6.68	0.98508	1.15908	3.5405

Finally, the performance of each model is analyzed on the RAM usage data from Google cluster data, and the specific results are shown in Table 3. Among the many comparative models, the traditional LSTM model is the one with the worst prediction accuracy, and the rest of the models have improved their prediction accuracy in comparison with it, among which the model in this paper has the most significant improvement. Compared with the LSTM model, the model in this paper has an improvement of up to 97.82% in the MSE evaluation index and 19.93% in the R^2 evaluation index.The TCN model is the model with the longest training and prediction time, while the model in this paper has the shortest training and prediction time. Compared to the TCN model, the training and prediction speed of this paper's model possesses an improvement of 40.71% and 22.86%, respectively.

Table 3. Performance comparison of models in RAM usage of Google cluster data

Model	MSE	R^2	Prediction time(s)	Training time(s)
LSTM	651.16	0.82555	1.73241	4.68104
BiLSTM	520.46	0.8765	4.88564	7.47698
TCN	549.26	0.88419	6.20389	15.36605
GRU	461.98	0.89625	1.09018	3.77337
Transformer	54.93	0.97672	1.29649	4.4748
Model of this article	14.17	0.99005	1.0272	3.61103

Through the comparison and analysis of the above multiple experiments, it can be learned that the supply chain inventory demand prediction model proposed in this paper performs well in terms of prediction accuracy and training and prediction time, and possesses excellent performance.

3. Supply Chain Inventory Demand Forecasting Application Practice

In this chapter, a practical study of supply chain inventory demand forecasting application will be conducted based on the proposed supply chain inventory demand forecasting model.

3.1. Data sets

The data of this applied practical research comes from the supply chain inventory data of an FMCG enterprise, and the experiments collected the sales of each product from January 2020 to December 2023, in which the inventory data from 2020 to 2022 is used as the model training set, and the sales data in 2023 is used as the model test set. And the data of order year, quarter and month, attributed province, belonging to the product specification, belonging to the price category, average monthly price, average monthly frequency, etc., which have been filtered by features, are combined to construct the dataset used for the inventory prediction in this paper. Meanwhile, in order to ensure the accuracy of the experiment, this dataset deletes the product data that has too little data during the period from 2020 to 2022. After data screening, 590 categories of product data and 407,352 order data were obtained as input data.

3.2. Experimental environment

The experimental environment CPU is AMD Ryzen 9 5900HX with Radeon Graphics, GPU is NVIDIA GTX3070 8G, 32G RAM, development environment is Pytorch 1.13.0+cu116, and Jupyternotebook platform is used.

3.3. Evaluation indicators

In this experiment, three types of criteria, mean square error (MSE), mean absolute error (MAE), and root mean square error (RMSE), are used to measure the performance comparison between this model and the benchmark model, respectively, and the smaller the values of MSE, MAE, and RMSE, the better the model prediction performance is represented.

3.4. Experimental results and analysis

Inventory data from January 2020-December 2022 are selected as training samples, and inventory data from January-December 2023 are used as test samples. The experiments all set the input sample step size as 36, and the prediction length of each dataset is divided into two groups of 3 and 12, and the specific parameters are shown in Table 4.

Table 4. Parameter setting of model

Parameters	Value
BatchSize	32
Learning rate	0.0005
Training rounds	200
Number of bulls	6
Number of hidden layers	16
Dropout	0.05
Optimizer	Adam

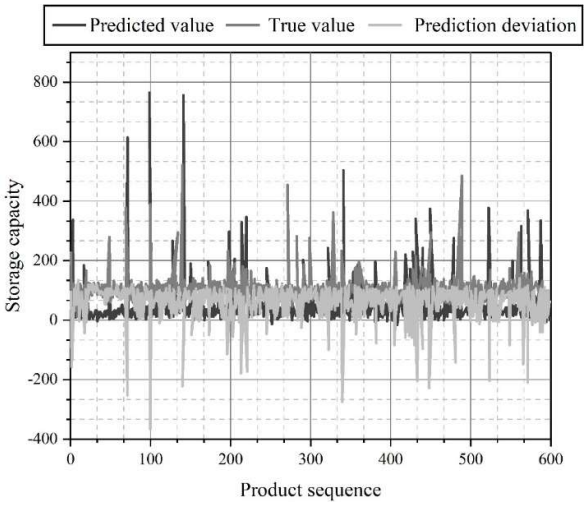
In this experiment, the VAR autoregressive model, LSTM model, and Transformer model are selected for comparison experiments with the supply chain inventory demand forecasting model proposed in this paper. The comparison of the prediction performance of this paper's model and other comparison models is shown in Table 5. It can be seen that this paper's model achieves the optimal performance in the three indicators of MSE, MAE, and RMSE. When the prediction length is 3, the values of MSE,

MAE, and RMSE indicators of this paper's model are 0.098, 0.082, and 0.288, respectively. When the prediction length is 12, the model in this paper still maintains good performance, with MSE, MAE, and RMSE index values of 0.202, 0.174, and 0.416, respectively. Obviously, the model in this paper possesses more stable long-term prediction performance. In addition, the prediction errors of this paper's model and the standard Transformer model are much smaller than those of the VAR and LSTM models, and marginally better than those of the standard Transformer model, which fully demonstrates the superiority of this paper's model in supply chain demand forecasting.

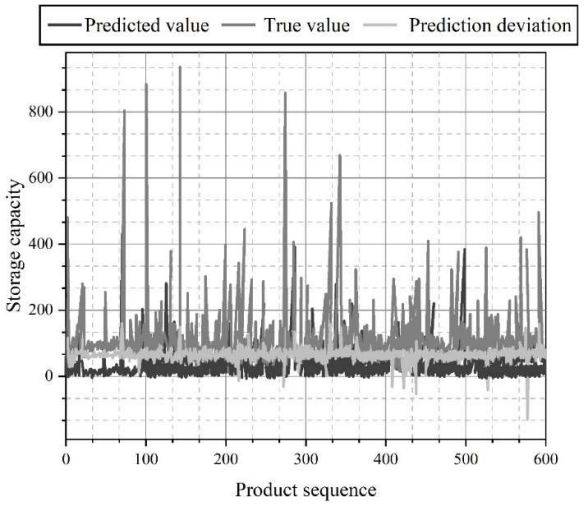
Table 5. Prediction results of each model

Prediction length	Evaluating indicator	VAR	LSTM	Transformer	Model of this article
3	MAE	0.199	0.204	0.125	0.098
	MSE	0.181	0.168	0.092	0.082
	RMSE	0.432	0.427	0.311	0.288
12	MAE	0.412	0.378	0.226	0.202
	MSE	0.406	0.361	0.206	0.174
	RMSE	0.734	0.603	0.446	0.416

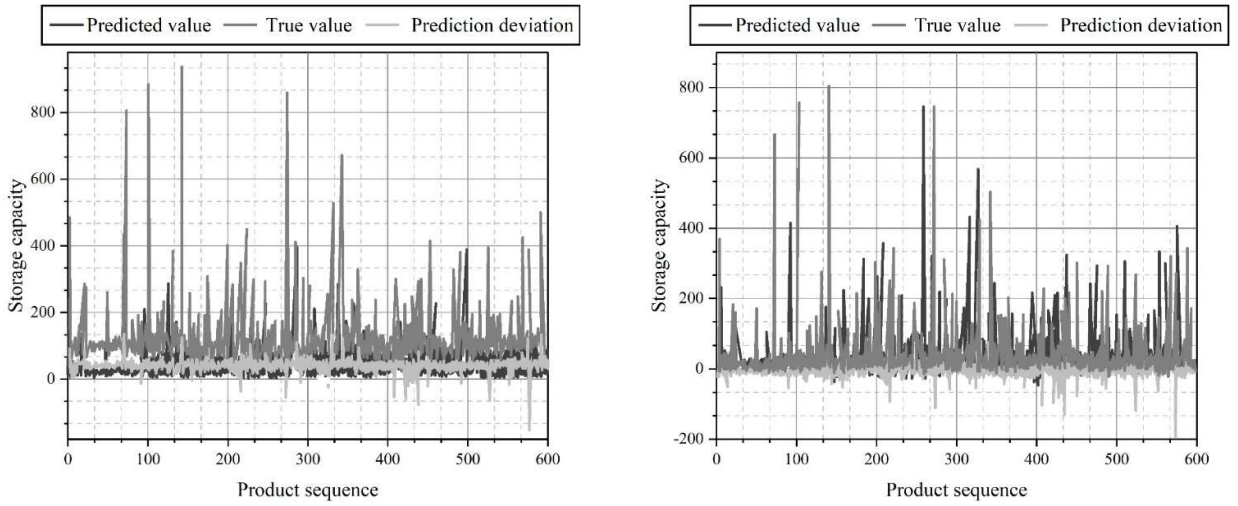
The results of the VAR model, LSTM model, Transformer model and this paper's model in the comparison between the prediction results and the real values of the inventory of different product supply chains are specifically shown in Figure 1. Figures (a) to (d) represent the VAR model, LSTM model, Transformer model and this paper's model in turn. It can be seen that the prediction deviation of this paper's model is extremely small, and the prediction curve is close to the true value. Among the other comparative models, the Transformer model can better predict the trend of the true value with smaller prediction deviation, while most of the experimental results of the LSTM model are far away from the error-tolerance interval.The experimental results of the VAR model show that the VAR model is unable to learn the relevant trend in this kind of problem, and the prediction results deviate greatly from the true value. Obviously, the model in this paper has achieved better results in the actual enterprise supply chain inventory demand forecasting, which verifies the effectiveness of the model in this paper.



(a)VAR



(b)LSTM



(c)Transformer (d)Model of this article

Fig. 1. Prediction deviation value

4. Nonlinear Optimization of Supply Chain Inventory Demand Forecasting

Supply chain inventory demand forecasting involves nonlinear factors such as seasonal fluctuations and market trends. Nonlinearity may lead to local multiple optimal solutions, which increases the accuracy of supply chain inventory demand forecasting and affects supply chain efficiency. In this chapter, a hybrid genetic algorithm (HGA) for solving nonlinear optimization problems will be proposed by combining the most rapid descent method and penalty function method with genetic algorithm.

4.1. Genetic Algorithms

Genetic algorithms (GAs) are derived from the study of in silico biological systems, which mimic chromosomal crossover and mutation processes in biological systems, and use a variety of different coding methods to represent feasible solutions to problems [32]. The GA algorithm is as follows: 1) Encoding: The solution data is set to be the manifestation of GA, and the process of mapping the phenotype to the genotype is called coding, and the gene string structure data is generated in the coding process, and the random combination of structural strings can form different points; 2) Generate the initial population: GA iterates with N initial point, sets the evolutionary algebra $t=0$, and the maximum number of iterations is Maxgen, generating the initial population with M individuals; 3) Evaluate $P(t)$ Fitness Values: Evaluate the fitness values of each individual: 4) Select: select offspring according to the roulette selection method: 5) Cross: Apply the cross operator P_c to the group; 6) Variation: Apply the mutation operator P_m to the population to obtain the next generation population $P(t+1)$;7) Judge the termination condition: when $t \le Maxgen$, $t++$, go to 3) step, otherwise, the individual with the maximum fitness value will be used as the optimal solution of the output and terminate the operation.

4.2. SHGA solving nonlinear unconstrained optimization problems

Specify the mathematical model of the unconstrained optimization problem (which will come with simple constraints):

$$\min f(X) \tag{11}$$

where $X=(x_1,\cdots,x_n)^T, X_1<X<X_2, X\in R^n$, the objective function $f(X)$ is continuously differentiable.

The HGA (SHGA) of the fastest descent method variant dieron combined with GA dieron, on the one hand, speeds up the convergence of the algorithm and prevents trapping in local optimality: on the other hand, it improves the accuracy of the optimal solution and produces the average adaptation value and the best individual with higher accuracy.

After initialization, the initial population is randomly generated, and the initial population contains M individual, the intermediate population containing N individuals (satisfying $N>2M$) is generated by using the roulette selection principle, the middle population is divided into sub-band population I and progeny population II according to w, the random point variation method and the fastest descent method are used to mutate the progeny population I and the progeny population II, respectively, and the fitness values of the individuals in the two populations are sorted, and the first M individuals are completely copied into the offspring to form a new population. Repeat the above process and stop as soon as the termination conditions are met.

The algorithm is described as follows.

1) Initialize each parameter to $Gen=0$.

2) Generate the initial population, which contains M individuals.

3) When $Gen>Maxgen$ iterations are terminated, output the result at the same time, or else go to the next step.

4) Generate an intermediate population of N individuals according to the roulette principle.

5) Divide the intermediate population into offspring population I and offspring population II, and mutate the offspring population I and offspring population II by random point mutation and fastest descent method respectively.

6) Sort the individuals of the population according to the fitness value from high to low;

7) The first M individuals are completely copied into the offspring to form a new offspring population, and order $Gen++$, go to step 3).

4.3. PHGA for nonlinear constrained problems

Mathematical models that specify nonlinear constraints:

$$\min f(X)\cdot s.t.\begin{cases} g_j(X)\le 0, j=1,2,3...,q \\ h_i(X)=0, i=q+1,q+2,...,m \\ a_i\le x_k\le b_i, k=1,...,n \end{cases} \tag{12}$$

Eq. $X=(x_1,\cdots,x_n), X\in R^n$. Where the constraints correspond to the inequality constraints, the equational constraints, and the constraint domains, respectively, and at least one of the objective function and the constraints is nonlinear. The mathematical model of Eq. (12) is simplified to a general nonlinear constraint model:

$$\min f(X).s.t.\begin{cases} g_j(X)\le 0, j=1,2,3...,q \\ h_i(X)=0, i=q+1,q+2,...,m \end{cases} \tag{13}$$

Take the adaptive value function as:

$$eval(X)=f(X)+pentlyf(X) \tag{14}$$

where $pentlyf(X)$ denotes the penalty term function. Form (15) is used for the penalty term $pentlyf(X)$ in (14):

$$penthyf(X)=\begin{cases}0 & \text{When } X \text{ is available}\\ \sum_{i=1}^{m} r_i g_i(X) & \text{other}\end{cases} \tag{15}$$

where r_i denotes the penalty coefficient for constraint $g_i(X)$, then the objective function is constructed:

$$F(X)=\begin{cases}f(X) & \text{When x is available}\\ f(X)+\sum_{i=1}^{m} r_i g_i(X) & \text{other}\end{cases} \tag{16}$$

The introduction of penalty twos can appropriately increase the non-feasible solutions for the next iteration, and at the same time, it reduces the infeasible solutions that severely violate the constraints, and the search process unfolds in the infeasible solution region. In this paper, we use the idea of outer-point penalty function, which is to gradually approximate the optimal solution from the infeasible solution region to the feasible solution, while the inner-point penalty function method requires the starting point to be in the feasible domain, so the outer-point penalty function method is more lenient. Incorporating the penalty function in the process of GA operation can improve the accuracy of the solution and facilitate the search for the global optimum.

The steps of HGA based on penalty function method are as follows.

1) Initialize each parameter to generate the initial population.

2) Make the number of iterations $Gen=0$ and calculate the objective function value $F(X)$.

3) Assign fitness value and perform selection, crossover, and mutation genetic operations on the current population $p(t)$.

4) Calculate the objective function value of the offspring, reinsert the offspring, $Gen++$.

5) Judge $Gen<Maxgen$, if it is not satisfied then output the minimum value as the result and terminate the operation, otherwise go to step 3).

5. Supply Chain Inventory Demand Forecasting Nonlinear Optimization Simulation Experiments

In this chapter, the effectiveness and optimal solution accuracy of the HGA algorithm proposed in this paper in the nonlinear optimization problem of supply chain inventory demand forecasting will be verified through simulation arithmetic cases and examples, respectively.

5.1. Nonlinear optimization simulation example analysis

The experimental functions are as follows:

$$\begin{gathered}\min_u f(x,u)=\frac{5000}{\frac{1}{12}u_1(x_1-2u_2)^3+\frac{1}{6}x_2u_2^3+2x_2u_2\left(\frac{x_1-u_2}{2}\right)^2}\\ g_1(x,u)=2x_2u_2+U_1(x_1-2u_2)\le 300\\ g_2(X,U)=\frac{180000x_1}{U_1(x_1-2u_2)^3+2x_2u_2\left[4u_2^2+3x_1(x_1-2u_2)\right]}+\frac{15000x_2}{(x_1-2u_2)u_1^3+2u_2x_2^3}\le 10\\ 10.0\le x_1\le 120.0\\ 10.0\le x_2\le 120.0\\ u_1\in[\underline{u}_1,\overline{u}_1]=[0.9,1.1]\\ u_2\in[\underline{u}_2,\overline{u}_2]=[1.8,2.2]\end{gathered} \tag{17}$$

In this algorithm, the specific parameters in the optimization function are set to $\xi = 0.0, \phi = 0.007, \psi = 0.0007$, the weight coefficient $\beta = 0.5$, the penalty factor $\sigma = 1000$, and the likelihood level of both uncertainty constraints are set to 0.9.

The HAG algorithm is partially described as follows.

1) Coding method: the two decision variables x_1 and x_2 are represented separately by 15-bit binary codes. From point 10 to 120, in turn, they are made to correspond to a binary coding string between 0 and 32767. The two binary coding strings representing x_1 and x_2 are then concatenated together and represented by a binary coding string of 30-bit length.

2) Decoding method: The formula for converting $y_i (i = 1, 2)$ to x_i with 15-bit binary codes y_1 and y_2 is:

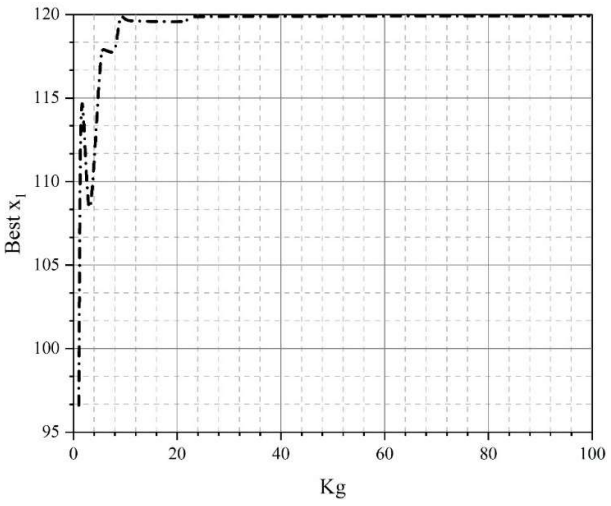
$$x_i = \frac{(120 - 10) \times y_i}{32767} + 10 \tag{18}$$

The optimization objective of the genetic algorithm is consistent with the objective function of the optimization problem, and the optimization objective is to find the minimum value of the function, so the value of individual fitness is taken as the inverse of the objective function.

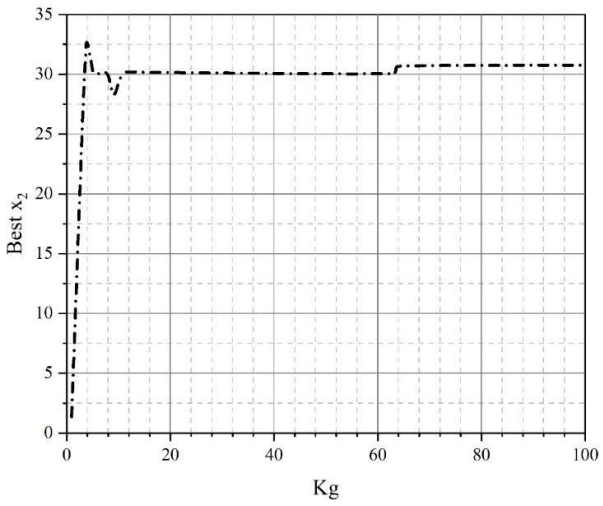
3) Genetic operators: proportional selection operator; single-point crossover operator; basic positional variation operator.

4) Genetic algorithm running parameters, termination of evolutionary generations is selected as 100, population size is selected as 100, mutation probability is selected as 0.1, crossover probability is selected as 0.5.

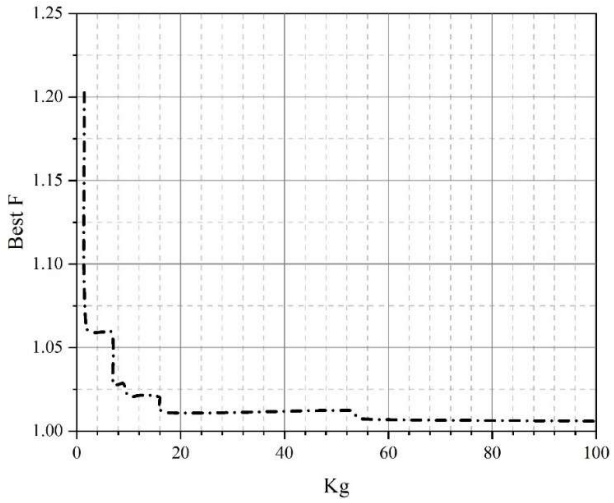
The single-objective evaluation function is 1.0065 and the optimization result of the objective function is 0.9933. The simulation results are shown in Fig. 2. Figures (a) to (c) represent the minimum optimization results of x_1, x_2 and the objective function in turn. The minimum optimization results of x_1 and x_2 all start to converge around 10, while the minimum optimization results of the objective function start to converge around 16. The convergence results are very good and get the expected results. Obviously, the HAG algorithm has a more stable convergence performance.



(a) x_1



(b) x_2



(c)Objective function

Fig. 2. Simulation result

5.2. Nonlinear Optimization Simulation Example Analysis

An unstable supply chain inventory demand forecasting system is known. Before the nonlinear optimization of supply chain inventory, the unit step response of the supply chain prediction result of this system is shown in Fig. 3. The unit step response presents an obvious fluctuation situation, and it can be seen from the unit step response situation that the supply chain prediction results of the system present an obvious unstable oscillating nonlinearity.

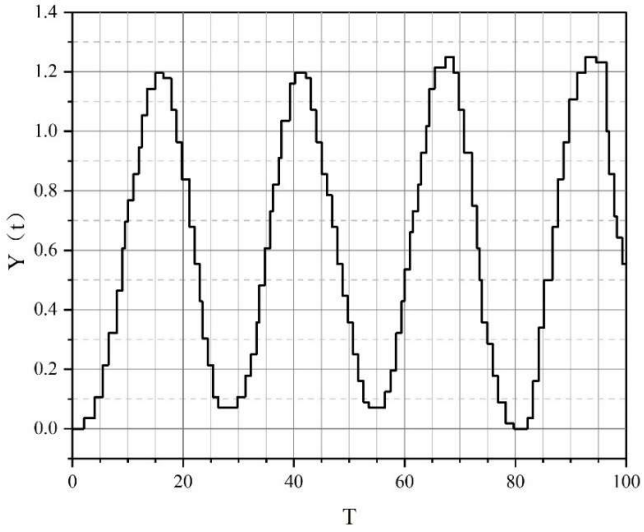


Fig. 3. Unit Step Response

Facing the nonlinear oscillations of the system's supply chain inventory demand forecasting results, the PID regulator is firstly used for correction.

Let the transfer function of the PID regulator be:

$$G_{PID(s)} = K_P \frac{1 + T_{IS} + T_D T_{IS}^2}{T_{IS}} \tag{19}$$

where: K_P, T_I, T_D are the optimization parameters of the PID regulator for proportional, integral and differential respectively. When simulated with MATLAB, the PID regulator is designed as an S-function

of the proportional, integral, and differential covariates, which facilitates the solution of the HAG algorithm.

The error signal is $e(t)=r(t)-y(t)$, and the objective function is taken to be the square integral of the error, i.e., $f(x)=\int e^2(t)dt$, where the upper limit of the integral, t, should be chosen to be large enough in terms of the regulation time, which is chosen to be 50s or 100s in this paper.According to the definition of the objective function and the magnitude of the error, the transformation from the objective function to the adapted value function is:

$$F(\underline{x})=\begin{cases}6-f(x) & \text{When } f(x)<6\\ 0 & \text{other}\end{cases} \tag{20}$$

The simulation is implemented by setting the PID regulator to be given optimized initial values of $K_P=0.63, T_I=0.0504\text{s}, T_D=1.9688\text{s}$. For this set of initial values, a unit step response is obtained, as shown in Fig. 4.

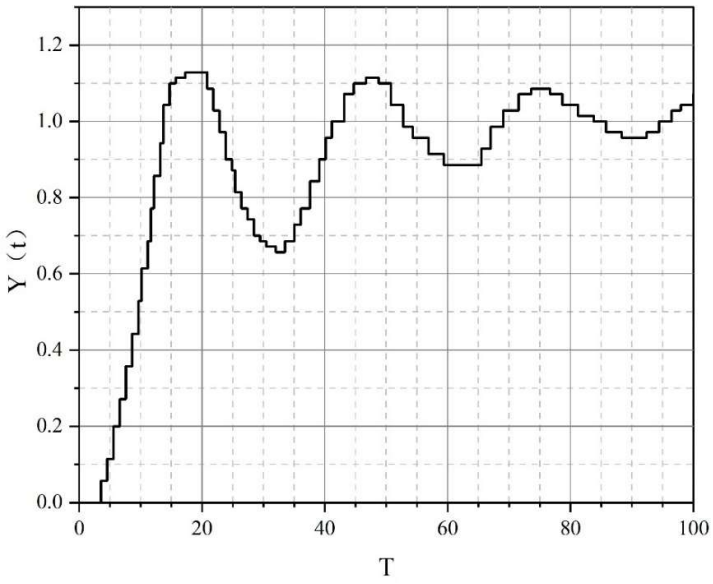


Fig. 4. Unit step response when the initial value is adjusted by PID

The HAG algorithm is used to set the population size N=10, the hybridization probability $P_c=0.25$, the mutation probability $P_m=0.001$, and the maximum number of generations $K=10$. Therefore, the optimal parameter value of the PID regulator can be obtained to be $K_P=3.6232, T_I=0.1175\text{s}, T_D=16.9950\text{s}$, and the unit step response at this time is shown in Fig. 5.

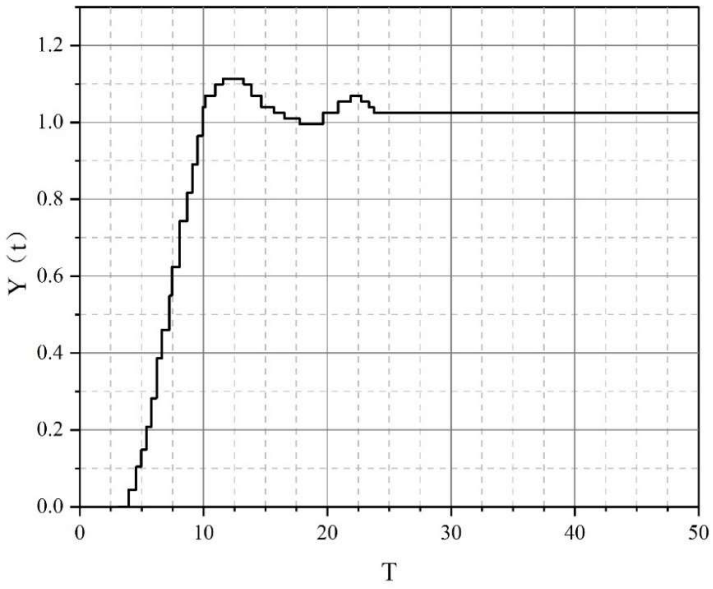


Fig. 5. Unit step response based on HAG algorithm

The traditional particle swarm algorithm (PSO) is chosen as a comparison algorithm and is applied for optimization with the optimal parameter of $K_P = 1.3353, T_I = 0.1547\text{s}, T_D = 8.3280\text{s}$. At this point, the unit step response is shown in Fig. 6.

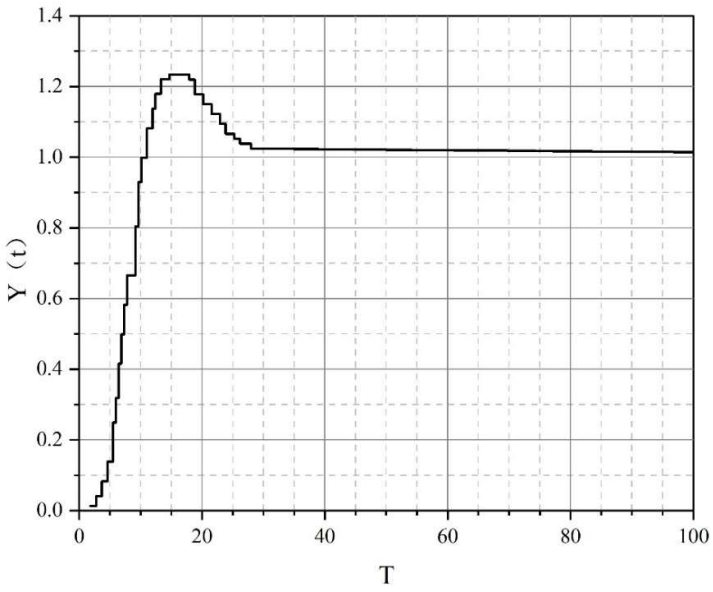


Fig. 6. Unit step response based on PSO algorithm

From the above simulation results, it can be seen that when the HAG algorithm is used for the nonlinear optimization of supply chain inventory, the overshooting amount of the supply chain inventory demand forecasting system is lower, the rise time is faster, the regulation time is shorter, and the results of the supply chain inventory demand forecasting are significantly improved. In contrast, when PSO algorithm is used for optimization, due to the nonlinear link, it is sometimes easy to fall into local optimum when searching for the optimum, and it occupies longer CPU time, more iterations, and the operation speed is slower. Therefore, for the supply chain inventory demand forecasting nonlinear optimization problem, to obtain the global optimum, it is appropriate to use the HAG algorithm to solve.

6. Conclusion

In this paper, the Transformer model in AIGC technology is chosen to construct a more adaptable supply chain inventory forecasting model, and hybrid genetic algorithm (HGA) is further proposed on the basis of traditional genetic algorithms as a solution to the nonlinear optimization problem faced by supply chain inventory forecasting.

Performance comparison experiments of supply chain inventory forecasting models are conducted using Google cluster count. In the load sequence, this paper's model of 1.10973s, 3.34986s, only slightly inferior to the GRU model, in the CPU usage data only the slowest and fastest GRU model in the training time is less than 1s, MSE, R^2 indicators performance are optimal. RAM usage data, this paper's model has the shortest training, prediction time, compared to the LSTM model in the MSE, R^2 evaluation indexes are 97.82% and 19.93% respectively. Comprehensively, the model in this paper shows excellent prediction performance.

The supply chain inventory data of an FMCG enterprise is used to carry out the application practice of supply chain inventory demand forecasting to test the practical application effect of this paper's supply chain inventory forecasting model. Compared with other models, this paper's model achieves optimal performance in MSE, MAE and RMSE. When the prediction length is 12, the model in this paper still maintains good performance, and the values of MSE, MAE, and RMSE indicators are 0.202, 0.174, and 0.416, respectively, which have more stable long-term prediction performance. For the prediction of supply chain inventory of different products, the prediction deviation of this paper's model is very small, the prediction curve is close to the real value, and the prediction performance is significantly better than the other comparative models.

The effectiveness of the HGA algorithm in the nonlinear optimization problem of supply chain inventory demand forecasting is verified through simulation cases and examples, respectively. In the simulation example analysis, the HAG algorithm shows more stable convergence performance. In the face of unstable nonlinear control system, after using HAG algorithm for supply chain inventory demand forecast optimization, the overshooting amount of supply chain inventory demand forecast system is lower, the rise time is faster, the regulation time is shorter, and the results of supply chain inventory demand forecast are significantly improved. Obviously, facing nonlinear optimization in supply chain inventory demand forecasting can be solved by HAG algorithm in this paper.

References

[1] Abolghasemi, M., Beh, E., Tarr, G., & Gerlach, R. (2020). Demand forecasting in supply chain: The impact of demand volatility in the presence of promotion. Computers & Industrial Engineering, 142, 106380.

[2] Tadayonrad, Y., & Ndiaye, A. B. (2023). A new key performance indicator model for demand forecasting in inventory management considering supply chain reliability and seasonality. Supply Chain Analytics, 3, 100026.

[3] Zhu, X., Ninh, A., Zhao, H., & Liu, Z. (2021). Demand forecasting with supply-chain information and machine learning: Evidence in the pharmaceutical industry. Production and Operations Management, 30(9), 3231-3252.

[4] Babai, M. Z., Boylan, J. E., & Rostami-Tabar, B. (2022). Demand forecasting in supply chains: a review of aggregation and hierarchical approaches. International Journal of Production Research, 60(1), 324-348.

[5] Ali, M. M., Babai, M. Z., Boylan, J. E., & Syntetos, A. A. (2017). Supply chain forecasting when information is not shared. European Journal of Operational Research, 260(3), 984-994.

[6] Pasupuleti, V., Thuraka, B., Kodete, C. S., & Malisetty, S. (2024). Enhancing supply chain agility and sustainability through machine learning: Optimization techniques for logistics and inventory management. Logistics, 8(3), 73.

[7] Singh, D., & Verma, A. (2018). Inventory management in supply chain. Materials Today: Proceedings, 5(2), 3867-3872.

[8] Shoushtari, F., Ghafourian, E., & Talebi, M. (2021). Improving performance of supply chain by applying artificial intelligence. International journal of industrial engineering and operational research, 3(1), 14-23.

[9] Boone, T., Ganeshan, R., Jain, A., & Sanders, N. R. (2019). Forecasting sales in the supply chain: Consumer analytics in the big data era. International journal of forecasting, 35(1), 170-180.

[10] Sharma, R., Shishodia, A., Gunasekaran, A., Min, H., & Munim, Z. H. (2022). The role of artificial intelligence in supply chain management: mapping the territory. International Journal of Production Research, 60(24), 7527-7550.

[11] Thomas, J., Vedi, V., & Gupta, S. (2021). Enhancing Supply Chain Resilience Through Cloud-Based SCM and Advanced Machine Learning: A Case Study of Logistics. J. Emerg. Technol. Innov. Res, 8(9), 357-364.

[12] Hofmann, E., & Rutschmann, E. (2018). Big data analytics and demand forecasting in supply chains: a conceptual analysis. The international journal of logistics management, 29(2), 739-766.

[13] Ramos, E., Pettit, T. J., Flanigan, M., Romero, L., & Huayta, K. (2020). Inventory management model based on lean supply chain to increase the service level in a distributor of automotive sector. Int. J. Supply Chain Manag, 9(2), 113-131.

[14] Chandriah, K. K., & Naraganahalli, R. V. (2021). RNN/LSTM with modified Adam optimizer in deep learning approach for automobile spare parts demand forecasting. Multimedia Tools and Applications, 80(17), 26145-26159.

[15] Fattah, J., Ezzine, L., Aman, Z., El Moussami, H., & Lachhab, A. (2018). Forecasting of demand using ARIMA model. International Journal of Engineering Business Management, 10, 1847979018808673.

[16] Huber, J., Gossmann, A., & Stuckenschmidt, H. (2017). Cluster-based hierarchical demand forecasting for perishable goods. Expert systems with applications, 76, 140-151.

[17] Hofmann, E. (2017). Big data and supply chain decisions: the impact of volume, variety and velocity properties on the bullwhip effect. International Journal of Production Research, 55(17), 5108-5126.

[18] Dash, R., McMurtrey, M., Rebman, C., & Kar, U. K. (2019). Application of artificial intelligence in automation of supply chain management. Journal of Strategic Innovation and Sustainability, 14(3).

[19] Oluwaseyi, J. A., Onifade, M. K., & Odeyinka, O. F. (2017). Evaluation of the role of inventory management in logistics chain of an organisation. LOGI–Scientific Journal on Transport and Logistics, 8(2), 1-11.

[20] Mostafa, N., Hamdy, W., & Alawady, H. (2019). Impacts of internet of things on supply chains: a framework for warehousing. Social sciences, 8(3), 84.

[21] Roßmann, B., Canzaniello, A., von der Gracht, H., & Hartmann, E. (2018). The future and social impact of Big Data Analytics in Supply Chain Management: Results from a Delphi study. Technological Forecasting and Social Change, 130, 135-149.

[22] Longo, F., Nicoletti, L., Padovano, A., d'Atri, G., & Forte, M. (2019). Blockchain-enabled supply chain: An experimental study. Computers & Industrial Engineering, 136, 57-69.

[23] Seyedan, M., & Mafakheri, F. (2020). Predictive big data analytics for supply chain demand forecasting: methods, applications, and research opportunities. Journal of Big Data, 7(1), 53.

[24] Zohra Benhamida, F., Kaddouri, O., Ouhrouche, T., Benaichouche, M., Casado-Mansilla, D., & López-de-Ipina, D. (2021). Demand forecasting tool for inventory control smart systems. Journal of Communications Software and Systems, 17(2), 185-196.

[25] Kilimci, Z. H., Akyuz, A. O., Uysal, M., Akyokus, S., Uysal, M. O., Atak Bulbul, B., & Ekmis, M. A. (2019). An improved demand forecasting model using deep learning approach and proposed decision integration strategy for supply chain. Complexity, 2019(1), 9067367.

[26] Feizabadi, J. (2022). Machine learning demand forecasting and supply chain performance. International Journal of Logistics Research and Applications, 25(2), 119-142.

[27] Deng, C., & Liu, Y. (2021). A Deep Learning-Based Inventory Management and Demand Prediction Optimization Method for Anomaly Detection. Wireless Communications and Mobile Computing, 2021(1), 9969357.

[28] Rensi Li. (2024). Research on Blended Teaching Model in Finance and Economics Education using AIGC Technology. Communication & Education Review(3).

[29] Shahriar Soudeep,Most. Lailun Nahar Aurthy,Jamin Rahman Jim,M.F. Mridha & Md Mohsin Kabir. (2024). Enhancing road traffic flow in sustainable cities through transformer models: Advancements and challenges. Sustainable Cities and Society105882-105882.

[30] Qi Li,Xiaoying Ren,Fei Zhang,Lu Gao & Bin Hao. (2024). A novel ultra-short-term wind power forecasting method based on TCN and Informer models. Computers and Electrical Engineering(PA), 109632-109632.

[31] Wei Jia,Ruizhe Ma,Li Yan,Weinan Niu & Zongmin Ma. (2025). Joint entity and relation extraction with table filling based on graph convolutional Networks. Expert Systems With Applications126130-126130.

[32] Sabahudin Vrtagic & Fatih Dogan. (2024). High-Frequency Gold Price Forecasting: Optimizing Multi-Layer Perceptron with Genetic Algorithm. Mathematical Modelling of Engineering Problems(12).