Computers & Industrial Engineering 177 (2023) 108998

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Learning to multi-vehicle cooperative bin packing problem via sequence-to-sequence policy network with deep reinforcement learning model

Ran Tian, Chunming Kang, Jiaming Bi, Zhongyu Ma [*], Yanxing Liu, Saisai Yang, Fangfang Li

Department of College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, PR China

ARTICLE INFO

Keywords:
Deep Reinforcement Learning
3D Bin Packing Policy
Position Sequence
Logistics Packing

ABSTRACT

In the logistics bin packing scenario with only rear bin doors, the packing sequence of items determines the utilization of vehicle packing space, but there is relatively little research on optimizing the packing sequence of items. Therefore, this article focuses on the bin packing sequence problem in the multi-vehicle cooperative bin packing problem(MVCBPP) and proposes a deep reinforcement learning model based on the sequence-to-sequence policy network with deep reinforcement learning model(S2SDRL). Firstly, the sequence-to-sequence neural networks model is constructed, which determines the packing probability of all items. The items will be packed by combining the bidirectional LSTM model and the attention module to construct the encoder and decoder. Secondly, the bin packing strategy of the items is obtained by the constructed reinforcement learning packing framework. Finally, the Seq2Seq policy network is updated and optimized by the policy gradient method with a baseline to obtain the current optimal packing strategy. In several bin packing scenarios, S2SDRL improves the average vehicle space utilization by more than 4.0% compared with the traditional packing algorithm, and the forward computation time of the model is much smaller than that of the traditional heuristic algorithm, so the model also has more realistic application value. Ablation experiments also confirm the effectiveness of the modules in the S2SDRL model for optimization of the packing order. The sensitivity analysis shows the model's some stability when the input data changes.

1. Introduction

One of the current challenges for logistics companies in the supply chain is bin packing, the packing problem is critical in improving vehicle space utilization and reducing transportation costs (Liang et al., 2020; Deng et al., 2019). In scenarios like pharmaceutical logistics with strict temperature control, the transport vehicle is usually a special temperature-controlled mini-van with only one compartment door. Using as few vans as possible is the key to saving labor and material resources since too many medicines must be transported to other urban areas, necessitating multiple small, strictly temperature-controlled vans, or cold chain vans. MVCBPP means that multiple vehicles are loaded together with the goods after collection, and the space utilization rate of all vehicles is approximately the same, aiming to reduce the number of containers.

Scholars used branch and bound Monte Carlo and other exact algorithms to solve the difficulties of single-vehicle loading, temporary loading, and container loading in the early period of the traditional packing optimization problem. The exact algorithm can effectively improve the solution efficiency. However, exact algorithm has high requirements for data, large memory space demand, and long operation time, and cannot obtain the optimal solution in a reasonable time. For the NP problem, the time required for the optimal solution exponentially increases with the size of the problem. Hence, researchers try to obtain the optimal solution to the bin packing problem using heuristic algorithms such as genetic algorithms(GA). Although the heuristic algorithms solve the problems of irregularly shaped goods container loading, multi-dimensional loading, and loading based on sequence transfer reduce the training time extremely and effectively avoids the case of trapping in a local optimum. Complex transformation and solution strategies are involved, and the speed of operation limits its actual application.

[*] Corresponding author.
E-mail addresses: tianran@nwnu.edu.cn (R. Tian), 2021222218@nwnu.edu.cn (C. Kang), mazybg@nwnu.edu.cn (Z. Ma), lyanxing@nwnu.edu.cn (Y. Liu), 2021222241@nwnu.edu.cn (S. Yang).

https://doi.org/10.1016/j.cie.2023.108998
Received 22 August 2022; Received in revised form 26 November 2022; Accepted 9 January 2023
Available online 12 January 2023

The availability of data in the supply chain channel and advances in machine learning methodologies offer new potential for solving supply chain issues (Zhu et al., 2021) and new ideas for bin packing problem. Researchers use DRL to solve multi-dimensional online and offline bin packing problems, which overcomes the shortcomings of heuristic methods such as long computing time, large memory occupation, and easily falling into a local-optimal. When the sequence is determined, the packing strategy determines the space position where the containers are loaded with goods. At the same time, the packing strategy also affects the generated sequence, making it very difficult to find a balance between them.

In the process of cooperative packing, it is important to determine the position sequence of packed items, but few scholars have focused on it. In this article, a DRL method based on position sequences is proposed for the 3D offline bin packing problem, which solves the problem of low space utilization and the long time to find the optimal solution of MVCBPP. The major contributions are as follows:

1. We constructed the sequence-to-sequence policy network with deep reinforcement learning model (S2SDRL) to reorder the items in the MVCBPP using the DRL agent. The Seq2Seq policy network outputs the packing order of the items, and the items are packed into the vehicle based on a layered bin packing strategy (LBPS) and a multi-vehicle cooperative packing strategy (MVCBPS), which extend the application of DRL on a single vehicle to multiple vehicles.
2. The agent in the Seq2Seq network's encoder and decoder module uses a bidirectional LSTM instead of a single direction LSTM to generate features that attribute one item with all the items above and below it, as well as describe the position sequence relationships of the items. The attention module is built between the encoder and decoder to highlight the key items' characteristics, effectively improving packing space utilization.
3. The most obvious finding to emerge from the analysis is that the S2SDRL method spends less time than the heuristic method in finding the optimal packing position sequence and has a higher average space utilization across multiple scenarios. And the total number of items can reach 1200 and more.

This article is structured as follows: Section 2 introduces the exact algorithm and the heuristic algorithm, analyzes the benefits and shortcomings of both, and reviews the progress of DRL algorithms on the packing problem; Section 3 defines the MVCBPP and adds constraints; Section 4 introduces the optimization model of packing sequence based on DRL, S2SDRL; Section 5 verifies that the proposed model has higher packing space utilization and shorter running time through experimental comparison with the heuristic algorithms, confirms the effectiveness of each module of the proposed model through ablation experiments, and also illustrates the stability of the model overall in response to changes in the data through sensitivity analysis; Section 6 concludes the article and discusses the related issues.

2. Related work

Machine learning(ML) benefits companies in the supply chain in both visible and invisible ways (Nagar et al., 2021). Delivery times are shortened, and supply chain effectiveness increases when operations are optimized (Liu et al., 2021). The risk management of each supply chain link (Schroeder and Lodemann, 2021; Sardar et al., 2021; Kosasih and Brintrup, 2022) lowers unreliability in the supply chain link. Financing banks and supply chain companies benefit equally (Liu and Hendalianpour, 2021) from the pricing analysis (Liu et al., 2021; Hendalianpour, 2020; Hendalianpour et al., 2020) of each link in the supply chain. One of the keys to lowering supply chain transportation costs and lowering the risk of unreliability in the transportation chain is to do an excellent job of packing items in logistics transportation.

Researchers have conducted on the packing problem and its variants. The solution has three main approaches: exact algorithms, heuristic algorithms, and machine learning algorithms. The current research status of these three types of algorithms will be introduced, respectively.

The exact algorithm is one of the earliest algorithms to solve the logistics packing problem, which mainly includes branch and bound algorithms (Elhedhli et al., 2019; Dell Amico et al., 2020), Integer programming algorithms (Baldi et al., 2019), linear programming algorithms (Martinovic et al., 2021; Liu et al., 2017), and dynamic programming (Bian et al., 2016). Although the branch and bound algorithm can effectively improve the solving efficiency, but branch and bound algorithm has high memory space and running time requirements. The linear programming algorithm requires higher data accuracy, only calculates linear problems with constraints, and is computationally intensive, significantly limiting the efficiency of solving nonlinear problems. The dynamic programming algorithm obtains the optimal value not only from the current state to the target state but also obtains the optimal value of the intermediate states, but it consumes more space.

The business of logistics transportation is becoming more and more complex, with different types, sizes, and weights of distributed goods, many distribution locations, and different transportation instruments, which form a logistics packing problem with multiple types of goods, multiple loading and unloading places, and multiple types of containers. This logistics packing problem with multiple types of goods, multiple loading and unloading places, and multiple types of containers involves too many variables, substantially increasing the problem's complexity. It is difficult to obtain the optimal solution in limited time.

Due to the time required to solve the optimal solution of the NP problem growing exponentially with the complexity of the problem size, various heuristic algorithms are attempted to obtain the optimal solution to the bin packing problem. The heuristic algorithms commonly used in the packing problem are the local search algorithm (Kramer et al., 2017; Abeysooriya et al., 2017), tabu search(Tabu) (Zazgornik et al., 2012; Crainic et al., 2009; Landero et al., 2020), simulated annealing(SA) (Yuan et al., 2108; Kämpke, 1988), variable neighborhood search (Santos et al., 2019), GA (Kucukyilmaz and Kiziloz, 2018) - (He and Wang, 2021), particle swarm optimization algorithm(PSO) (Fang et al., 2021) ant colony optimization algorithm(ACO) (Levine and Ducatelle, 2004). In the related research, iterative search has a higher probability of searching for the optimal global solution by multiple iterations. Still, searching for the effective optimal solution is problematic because it is mixed with disturbance factors. At the same time, the disturbance is too large or too small, which will have affect the iterative results differently. Local search is searching a local region and its neighbors, trapped easily in a local optimum. Tabu search can search different paths but involves complex neighbor transformation and solution strategy, which is not easily implemented in the reality of logistics business. A simulated annealing algorithm has better results for packing problems. However, simulated annealing algorithm can often only give an approximate optimal solution due to the limitation of computation time in actual applications. The variable neighborhood search algorithm is still an algorithm that breaks out of the local optimum, different from the tabu search and simulated annealing algorithms, this algorithm does not follow a certain 'trajectory', but it is still tough to get the global optimal solution. The GA has its shortcomings, such as easy convergence and low local searchability, so it cannot guarantee that the global optimum is converged. The particle swarm and ant colony optimization algorithms have the advantages of fast convergence, low dependence on experienced parameters and simplicity, etc. However, there are shortcomings, such as being easily trapped in local optimum and long computation time as other heuristic algorithms. Although the heuristic algorithm can avoid trapping in local optimum, which involves complex transformation and solution strategies, the computational speed is the main factor limiting its application in reality.

In order to solve the above problems, more and more researchers attempt to solve bin packing problems with machine learning. They use machine learning to improve heuristic algorithms or use machine learning directly to solve bin packing problems. Kasap and Agarwal

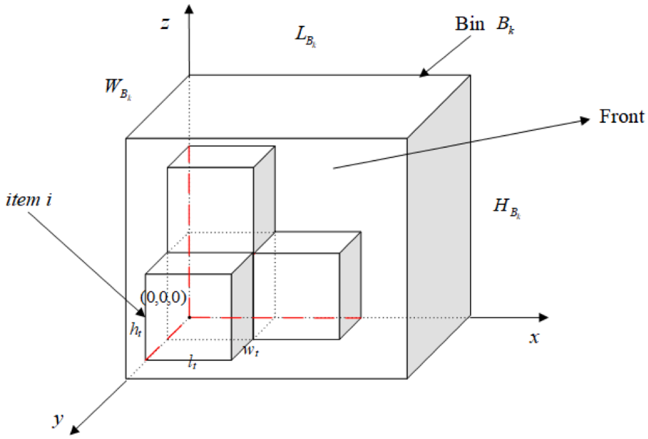

Fig. 1. items packed into a bin.

(2012) first used neural networks to solve the classical packing problem, and the algorithm can solve most baseline problems with optimal or approximately optimal solutions. Duan et al. (1804) used a multi-task selected learning approach to solve the 3D flexible packing problem well. Deep learning has a more vital perceptual ability but cannot make some decisions. In contrast, reinforcement learning has a more vital decision ability but it is helpless with perceptual problems, so effectively combining them can provide a good solution for complex packing problems. Hu et al. (2017) obtained better results than heuristic algorithms by applying DRL to the 3D bin packing problem the first time. Kundu et al. (2019) proposes a packing algorithm based on the Double DQN framework by studying the two-dimensional bin packing problem from the perspective of state feature extraction. Tijjani and Bucak (2013) used the Q-Learning algorithm to solve the container loading problem and obtained a enormous container load with less space to be wasted. Saikia et al. (2018) combines evolutionary strategies and reinforcement learning methods to obtain similarly better results than heuristic algorithms in obtaining the optimal sequence of packing problems. Zhao et al. (2006), Zhao et al. 2018) proposed a method that could reach the human level of packing space utilization by combining the framework of Actor-Critic. Peters and Schyns (2020) proposed to combine AR with reinforcement learning methods to solve the packing problem with many packages. Hu et al. (2020) proposed a DRL algorithm that could compactly pack the item into the bin. Verma et al. (2007) solved the three-dimensional bin packing problem (3DBPP), and experimentally demonstrated that the performance of this method is similar to the human level. Zhang et al. (2017) proposed a self-attention-based RL method that surpassed other traditional methods in space utilization. Jiang et al. (2021) combined DRL with planning constraints to solve the larger-scale bin packing problem.

In the MVCBPP, each vehicle is loaded by the position sequence and packing strategy, so it is essential to identify the position sequence. Still, only some scholars have focused on the packing sequence of this problem. In this article, based on the DRL method, the Seq2Seq network is used to make sequence decisions, and the strategy gradient method based on the baseline is used to optimize the model. The S2SDRL is proposed to solve the problem of low space utilization based on the position sequence of vehicle cooperative packing. Our managers must assess whether the solution is feasible while enhancing the privacy and protection of software data after using machine learning methods for judgment (Nayal et al., 2021).

3. Problem definition

Assuming that the bins and the items are rectangular, a bin B_k will be packed with items, the bin's length is$L$, the width is$W$, the height is$H$, sizes of all other bins are the same. A series of items will be loaded, the quantity of items isN, the length is$l$, the width is$w$, and the height is$h$, the i-th item can be represented as(l_i, w_i, h_i). Now the i-th item is packed into k-th binB_k, as shown in Fig. 1.

Table 1
Definition of symbols.

Symbol	Significance
I	set of N items
l_i, w_i, h_i, v_i	length, width, height, and volume of the i-th item
B	set of all bins
B_k	the k-th bin in the set B of all bins
L_k, W_k, H_k, V_k	length, width, height, and volume of the k-th bin
L_t, W_t, H_t, V_t	length, width, height, and volume of the packed bin
l_t, w_t, h_t	the length, width, and height of the item when temporarily packed
quantity	number of items already packed in one bin
CL	set of items already packed in one bin
CLA	a set of packed bin set *CL*
P	set of items priority packing probabilities
PL	the probability of priority packing of the item *P*
PLA	set of *PL*
PosSeq	set of position sequences of items
IRPS	set of items repacked after combining position sequences
IRP	set of probabilities repacked after combining position sequences
RLS	set of space utilization of all bins
RD	discount values for all bins space utilization set
r	packing space utilization of one single bin
R	the set of *r*
quantities	a set that records the number of items packed in all bins

Table 2
Variable definitions of constraints in the single bin packing process.

Variable	Significance
x_i	the position of the item i on the x-axis
y_i	the position of the item i on the y-axis
z_i	the position of the item i on the z-axis
$left_{ij}$	the item i is on the left side of the item j
top_{ij}	the item i is on the top side of the item j
$behind_{ij}$	the item i is on the behind side of the item j
δ_{i1}	the frontal orientation of the item i is the x-positive direction of the bin
δ_{i2}	the frontal orientation of the item i is the x-negative direction of the bin
δ_{i3}	the frontal orientation of the item i is the y- positive direction of the bin
δ_{i4}	the frontal orientation of the item i is the y-negative direction of the bin
δ_{i5}	the frontal orientation of the item i is the z- positive direction of the bin
δ_{i6}	the frontal orientation of the item i is the z-negative direction of the bin

For the bin packing problem based on the packing position sequence, our goal is to find an efficient packing sequence and strategy packs the items (l_i, w_i, h_i) into M bins of the same size according to the packing position sequence and to maximize the average packing space utilization $\overline{R}$ of all bins. The goal function is denoted as follows:

$$\max(\overline{R}) = \max\left(\frac{1}{M}\sum_{k=1}^{M}\frac{\sum_{i}^{quantities_k} l_i \times w_i \times h_i}{L_k \times W_k \times H_k}\right) \tag{1}$$

Where $quantities_k$ denotes the number of items packed into the k-th bin. The symbols involved in this article and the meanings represented are shown in Table 1.

We will define the variables related to the packing constraints in Table 2.

We combining Fig. 1 and Table 2, the defined constraints are as follows.

$$\begin{cases} left_{ij}, top_{ij}, behind_{ij} \in \{0, 1\} \\ \delta_{i1}, \delta_{i2}, \delta_{i3}, \delta_{i4}, \delta_{i5}, \delta_{i6} \in \{0, 1\} \end{cases} \tag{2}$$

The point of origin is$(0, 0, 0)$, where $left_{ij} = 1$ means that item i is to the left of item$j$, $top_{ij} = 1$ means that item i is below item$j$, and $behind_{ij} = 1$ means that item i is behind item$j$. $\delta_{i1} = 1$ means that the front side of the item i is oriented to the positive axis of the x-axis of the bin. $\delta_{i2} = 1$ means that the front side of the item i is oriented to the negative axis of the x-axis of the bin. $\delta_{i3} = 1$ means that the front side of the item i is

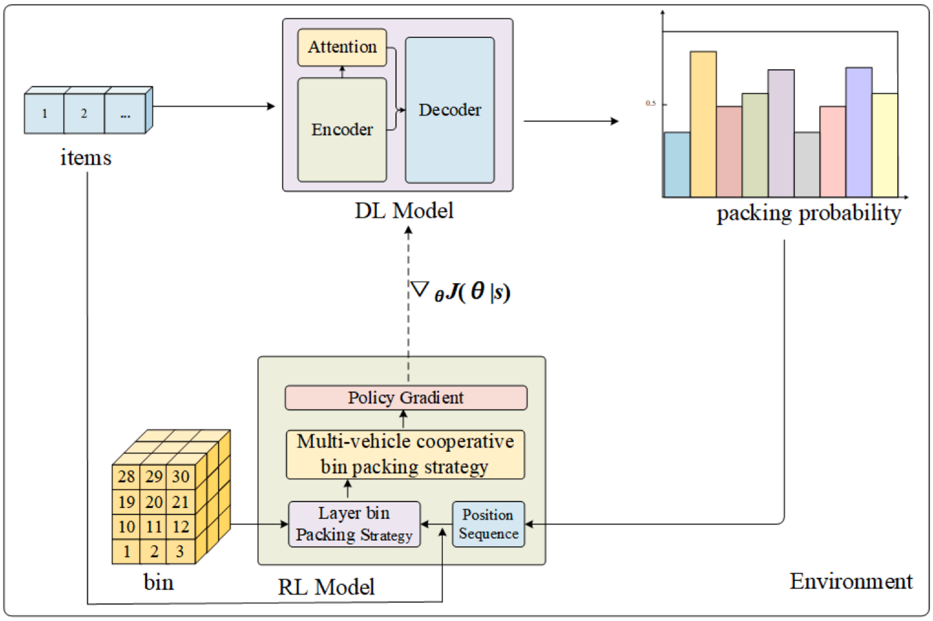

Fig. 2. DRL-based packing sequence optimization framework.

oriented to the positive axis of the y-axis of the bin. $\delta_{i4} = 1$ means that the front side of the item i is oriented to the negative axis of the y-axis of the bin. $\delta_{i5} = 1$ means that the front side of the item i is oriented to the positive axis of the z-axis of the bin. $\delta_{i6} = 1$ means that the front side of the item i is oriented to the negative axis of the z-axis of the bin. When the orientation of the item is determined, the length, width, and height of the item are denoted as(l_i^*, w_i^*, h_i^*). Formula (2) denotes the uniqueness of the item orientation and the uniqueness of the packing direction of the item.

$$\begin{cases} x_i - x_j + L_k \times left_{ij} \leqslant L_k - l_i^* \\ y_i - y_j + W_k \times top_{ij} \leqslant W_k - w_i^* \\ z_i - z_j + H_k \times behind_{ij} \leqslant H_k - h_i^* \end{cases} \tag{3}$$

Formula (3) denotes that the length, width, and height of the items packed cannot overlap with each other.

$$\begin{cases} 0 \leqslant x_i \leqslant L_k - l_i^* \\ 0 \leqslant y_i \leqslant W_k - w_i^* \\ 0 \leqslant z_i \leqslant H_k - h_i^* \end{cases} \tag{4}$$

Formula (4) denotes that the bin can contain the length, width and height of the item packed.

$$\begin{cases} l_i^* = \delta_{i1} l_i + \delta_{i2} l_i + \delta_{i3} w_i + \delta_{i4} w_i + \delta_{i5} h_i + \delta_{i6} h_i \\ w_i^* = \delta_{i1} w_i + \delta_{i2} h_i + \delta_{i3} l_i + \delta_{i4} h_i + \delta_{i5} l_i + \delta_{i6} w_i \\ h_i^* = \delta_{i1} h_i + \delta_{i2} w_i + \delta_{i3} h_i + \delta_{i4} l_i + \delta_{i5} w_i + \delta_{i6} l_i \end{cases} \tag{5}$$

Formula (5) denotes the new length, width and height obtained by turning the item according to the orientation.

$$\begin{cases} left_{ij} + top_{ij} + behind_{ij} = 1 \\ \delta_{i1} + \delta_{i2} + \delta_{i3} + \delta_{i4} + \delta_{i5} + \delta_{i6} = 1 \end{cases} \tag{6}$$

Formula (6) denotes that item i is to the left, above or behind the item j and the item' orientation can only be one of the six types of item orientation.

4. The packing sequence optimization model based on DRL

In order to solve the 3D packing problem based on packing sequence optimization, we build a deep reinforcement learning framework, which

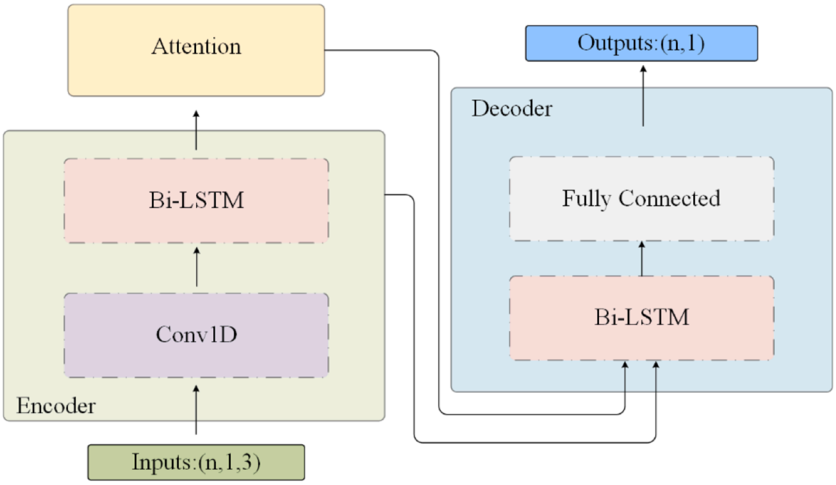

Fig. 3. The Seq2Seq network model based on Bi-LSTM.

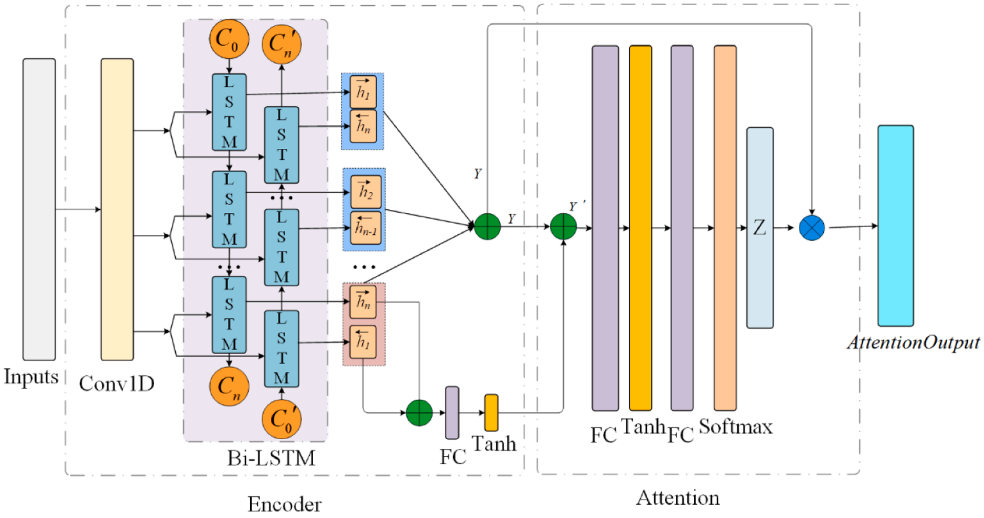

Fig. 4. The structure of the encoder.

includes three parts: Seq2Seq network, layered bin packing strategy (LBPS), and reinforcement learning optimization method based on policy gradient. The following is the description of these modules separately.

4.1. DRL-based packing sequence optimization framework

Most of the studies on the three-dimensional bin packing problem aim to find the packing position (x, y, z) to be packed of the items. Each packed item has two characteristics: (1) the item's length, width, and height. (2) the order of the item in the packing position sequence.

As shown in Fig. 2, the DRL-based packing sequence optimization framework consists of four parts: items, bins, deep learning (DL), and reinforcement learning (RL). The agent feeds the items' length, width, and height into the DL model to calculate the packing probability. RL model packs all of the items into multiple bins using the packing probability and item's length, width, and height then calculates the gradient of the DL model using the policy gradient update formula with baseline$\nabla_{\theta}J(\theta|s)$. Then the DL policy network updates the network parameters based on the gradient calculated by the RL model. The Seq2Seq network, which includes Encoder, Attention, and Decoder, is used to build the DL model. The RL model consists of four parts: position sequence, layered bin packing strategy, multi-vehicle cooperative bin packing strategy, and policy gradient.

4.2. The Seq2Seq policy network based on Bi-LSTM

The packing sequence probabilities in the packing sequence optimization model based on the DRL method are predicted by the Seq2Seq model based on the Bi-LSTM. The Seq2Seq model plays a policy role in the reinforcement learning process, which could be called a policy network. The model combines a convolutional neural network(CNN), Bi-LSTM module, and attention to obtain the packing probability of all the items waiting to be loaded. The network structure framework is depicted in Fig. 3.

The structure of the encoder(Encoder) based on the Bi-LSTM network model is depicted in Fig. 4.

The N items to be packed $((l_1, w_1, h_1), (l_2, w_2, h_2), ..., (l_N, w_N, h_N))$ are rewritten as$I = (x_1, x_2, ..., x_N)$, x_i represents the information of the current item to be packed, I represents the set of all items. All items with position sequences in I are input to the convolution module of the encoder. The output value of the item is shown as $e_{i,m}$ in formula (7) after the i-th item is processed by a one-dimensional convolution with m convolution kernels.

$$e_{i,m} = \sum_{j=0}^{channel_{in}-1} W_{m,j} \otimes x_{i,j} + b_m, (m \in (1, 2, ..., channel_{out})) \tag{7}$$

Where $channel_{in}$ is the channel number of the input data, $channel_{out}$ is the channel number of the output data, $\otimes$ is the cross-correlation operator. All items' length, width, and height are sent into the one-dimensional convolution module (Conv1D). This module processes the features using multiple convolution kernels of size one. Then, this module transposes the output, which becomes an up-dimensioning operation to each feature, representing each feature as a vector. As a result, the Conv1D can be seen as an embedding process. The Bi-LSTM may obtain the correlation relationship before and after the pack after Conv1D extracts the information of each packed item. The positive formula of the Bi-LSTM is as follows.

$$\widetilde{C}_i = \tanh(W_C \cdot [h_{i-1}, e_i] + b_C) \tag{8}$$

which is a linear activation of short term memory on one lstm cell.

$$C_i = \sigma\left(W_{fg} \cdot [h_{i-1}, e_i] + b_{fg}\right) \times C_{i-1} + \sigma\left(W_{ug} \cdot [h_{i-1}, e_i] + b_{ug}\right) \times \widetilde{C}_i \tag{9}$$

which belong to forgetting gate and update gate on one lstm cell.

$$y_i = h_i = \sigma\left(W_{og} \cdot [h_{i-1}, e_i] + b_{og}\right) \times \tanh(C_i) \tag{10}$$

which belong to output gate on one lstm cell. e_i denotes the feature of the current item i after one-dimensional convolutional processing, y_i denotes the information related to the current item i and the packed item (the initial state $y_0 = 0$), W and b are the parameters of the Seq2Seq network model. The activation function σ of each LSTM cell's forgetting gate, update gate, and output gate is Sigmoid function.

The negative computation only requires inputting the positive input data into the LSTM module in reverse order, and h_i' denotes the hidden output of each loop. Positive and negative outputs of each hidden layer are concatenated to obtain$Y$. The initial and ending values of long-time memory for the positive LSTM are denoted as c_0 andc_n, and the initial and ending values of long-time memory for the negative LSTM are denoted as c_0' andc_n'. From Fig. 4, h_n and h_n' are the values obtained from the last loop of Bi-LSTM; the concatenated value of both are calculated by the fully connected (FC) layer and activated by the Tanh function, then concatenated with$Y$, and calculated Z by the two FC layers.Z, as the attention, is multiplied by Y to get the attention's output(AttentionOutput), which is also the input of the decoder.

AttentionOutput is used as input to the decoder part. Inputs are calculated by inputting AttentionOutput into the Bi-LSTM module of the

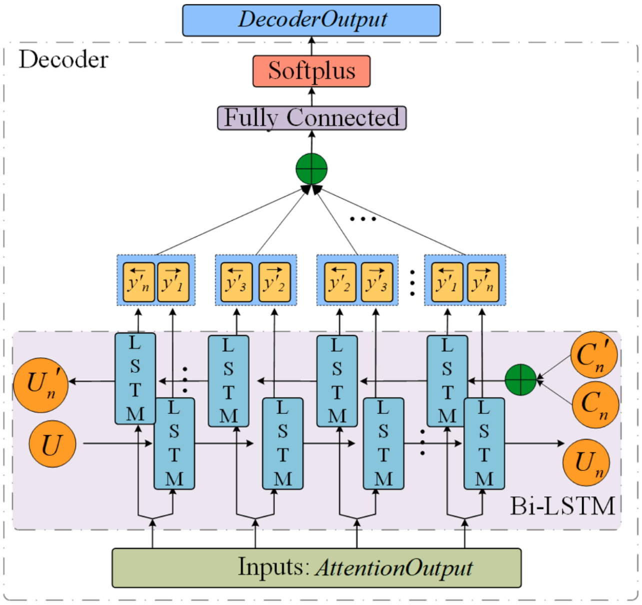
DecoderOutput
Decoder
Softplus
Fully Connected
$\overleftarrow{y'}_n$ $\overrightarrow{y'}_1$
$\overleftarrow{y'}_3$ $\overrightarrow{y'}_2$
$\overleftarrow{y'}_2$ $\overrightarrow{y'}_3$
$\overleftarrow{y'}_1$ $\overrightarrow{y'}_n$
LSTM
U'_n
U
C'_n
C_n
U_n
Bi-LSTM
Inputs: AttentionOutput

Fig. 5. The structure of the decoder.

decoder. The positive LSTM uses S as the initial long-time memory, and the negative LSTM concatenates c_n and $c_n^{'}$ in the dimension of the feature as the initial long-time memory. Furthermore, calculated a FC layer to obtain the initial packing probability of the items. The Softplus activation function is introduced to solve the problems that FC layer output contains the non-positive value, which maps the packing probability values of all items to non-negative numbers. The Softplus function formula is as follows.

$$softplus(x) = \log(1 + \exp(x)) \tag{11}$$

As depicted in Fig. 5, the Softplus activation function calculates the decoder output (DecoderOutput), which is also the output of the whole Seq2Seq policy network.

4.3. Position sequence optimal strategy

The packing probability (Loading Probability) is a policy network output used by the agent to sort and pack the items and update the policy network based on the packing results. The agent can obtain the packing position sequence of all items in descending order, pack the items into the bins in order by the sequence until all the items are packed, then obtain the list of packing probability of items loaded in each bin (Probability List P), the set of Reward List RLS for each bin, and calculate the discounted value of each bin (Discounted Reward List RD). Agent optimizes the policy network by computing the policy gradient $\nabla_\theta J(\theta|s)$ to improve the average packing space utilization of the bin. The RL method trains the policy network to optimize the item packing sequence in the MVCBPP. The RL process is depicted in Fig. 6.

The DRL model uses the seq2seq network model as the policy networkπ. All items' length, width, and height are inputted into the policy network. The policy network outputs$\pi_\theta(\cdot|x)$, and θ denotes the parameters of the network model. Each item requires a packing action to be packed into the bin, and $\pi_\theta(\cdot|x)$ is the set A of all actions to pack all items, with the length of A equal to the number of items. We define that

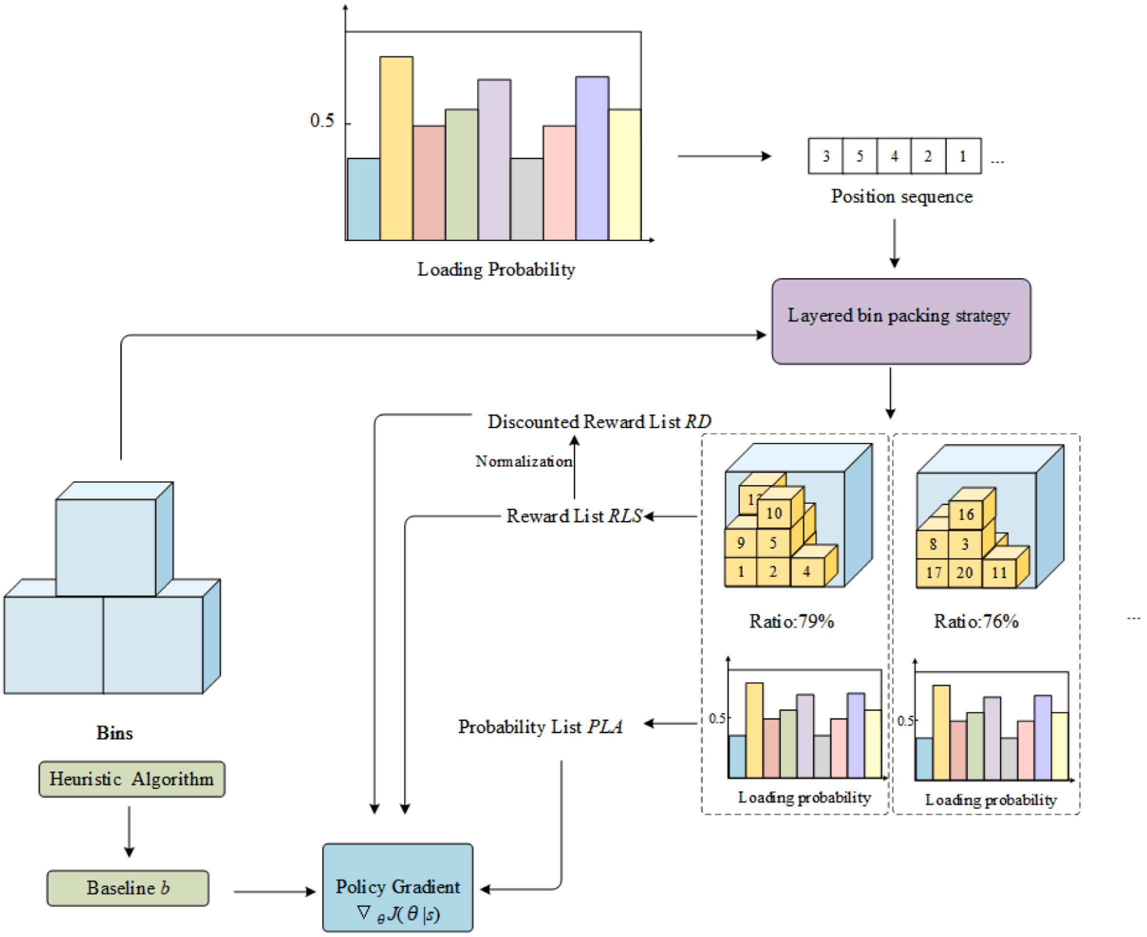

Fig. 6. RL process.

if$a_i \geqslant a_j, (i > j \text{and} a_i, a_j \in A)$, the item x_i mapping to a_i is packed first. The elements of A are arranged in descending order to get the set of packing sequence*PosSeq*, and *PosSeq* is the set of items that records the order of packing items into bins.

During the training process, the agent uses a policy learning method with a baseline. After interacting with the environment, the agent gets the set of rewards*RLS*, the set of item quantities*quantities*, the probability of items packing*PLA*, the information of packed items *CLA* for each bin, and the interaction process between the agent and the environment is stated as follows. The initial state of all bins is no item. The initial environment state is the scenario where all bins are empty, which is noted asS_0. The items in the *PS* are packed into several bins in sequence; the environmental state is recorded as S_1 when x_0 is packed intoB_0. The reward is the total volume of all items inB_0, recorded as$r_1 = v_1$. The state changes to S_2 after x_1 is packed into the B_0 by the LBPS,$r_2 = v_1 + v_2$. The items are packed into B_0 in turn by the LBPS until B_0 cannot be packed with next item, the reward is $r_i = \sum_{u=0}^{i} v_u$ when a item being packed. $q(q = \{1, 2, ..., n\})$ denotes the *q-th* item that has been packed in the *k-th* bin, the reward for the *q*-th item of B_k is denoted as

$$r_{k,q} = \frac{v_u}{V_k}, u = quantities_{k-1} + quantities_{k-2} + ... \tag{12}$$

The LBPS and the vehicle cooperative packing strategy pack all items and calculate the reward for each item; the reward for all items consists of a set that$RLS_{k,q} = r_{k,q}$. The accumulated discount reward for the *j-th* item of B_k is denoted as$RD_{k,q}$, which is obtained from formula (23).

$$RD_{k,q} = r_{k,q} + \sum_{u=quantities_{k-1}+quantities_{k-2}+...}^{q+quantities_{k-1}+quantities_{k-2}+...} \gamma \cdot RLS_{k,q+1} \tag{13}$$

In the MVCBPP, the agent decides to get all packing position sequences. After packing items into the bins, each item's actions, states, and rewards consist of a Markov decision process(MDP), denoted as τ.

$$\tau = (S_0, A_1, r_1, S_1, A_2, r_2, S_2 ...) \tag{14}$$

The score of the environmental state after packing x_i is indicated by the state-value function $V_{\pi_\theta}(S_i)$. The objective function of the policy gradient method is defined as:

$$J(\theta|s) = E_{o \sim \pi_\theta(\cdot|x)}\left[V_{\pi_\theta}(S)\right] \tag{15}$$

In this article, the gradient is calculated in the following way: in each training step, to obtain the expected value of the state value, after packing all the items into the bins using the LBPS and the MVCBPS, the gradient of the probability of packing the items and the corresponding discounted value of the goods for all the carriages in the current state is calculated, and this gradient is used to determine the expected value of the state value.

$$\nabla_\theta J(\theta|s) = \left(\sum_{k=1}^{M}\left(\sum_{q=1}^{n} RD_{k,q} \times \log PLA_{k,q}\right) \times (b_k - RLS_k)\right) + \left(b_v - \frac{1}{M}\sum_{k=1}^{M} RLS_k\right) \tag{16}$$

In formula (16), $k(k = \{1, 2, ..., M\})$denotes the k-th bin, M denotes the number of bins. b_v is the new baseline value after the v-th update of the network.

In order to initialize a relatively reasonable baseline value b_0 and avoid using an unreasonable baseline value which will increase the model training time, we solve the MVCBPP with the random search algorithm, simulated annealing algorithm, ant colony optimization algorithm, GA, particle swarm optimization algorithm, differential evolution algorithm, and Tabu search algorithm, calculate the average of the space utilization of these six heuristics in the four scenarios respectively. The update formula for the baseline b_v is defined as

$$b_v = b_{v-1} + \lambda\left(b_{v-1} - \frac{1}{M}\sum_{k=1}^{M} RLS_k\right) \tag{17}$$

Adam Optimizer updates all parameters of the policy network θ.

$$\theta = \text{Adam}(\theta, \nabla_\theta J(\theta|s)) \tag{18}$$

The above equation can be expressed using $\theta_t = \theta_t - AdamLr \times \text{Adam}_t(\nabla_\theta J(\theta|s))$ when updating the network parameters at each step.$AdamLr = 0.001$. $\text{Adam}(\nabla_\theta J(\theta|s))$ is the value after the gradient is optimized by the Adam algorithm.

With the above symbols defined, we give the overall DRL training process, as described in Algorithm 1.

```
Algorithm 1: DRL training process
Input: The set of all items$I$.
Output: The best bin packing position sequence$PS$.
Initialize: The number of samples for training isN, initialize the training step$T$,
calculate and obtain the initial packing rate baseline$b_0$.
for$j = 1, 2, ..., T$:
Input $I$ into the policy network, get the action set$A$, arrange the element of
$A$ in descending order to obtain$PS$, get$quantities$,$PLA$,$RLS$,$CLA$,$R$,$RD$
Through the LBPS and MVCBPS.
The $q$-th($q = 1, 2, ..., q \in CC_k$) item of the $k$-th bin is selected in turn to
calculate the gradient: formula (16)
Update baseline: formula (17)
Update network parameters: formula (18)
end for
return the best $PS$
```

4.4. Layered bin packing strategy

A packing strategy is proposed for each item $x_i \in I$ to be packed into a bin B_k based on the layered idea. During packing a single bin, there are three significant matrices to operate: item$x_i = (l_i, w_i, h_i)$, the bin packed space (L_t, W_t, H_t, V_t) after packing an item, and the maximum length, width, and height (l_t, w_t, h_t) of all packed items in one layer. The LBPS has three main steps.

Firstly, a $L_t = W_t = H_t = 0$ temporary bin is created. $CL = \varnothing$ denotes the set of items temporarily packed in this bin, and $c = 0$ denotes the number of items. When the items are not packed into the vehicle, the temporary parking space's length, width, and height are$l_t = w_t = h_t = 0$.

Secondly, when$i = 1$, if the item is not packed with the bin and the packing conditions are met, the first item in the current packing sequence is packed into the bin, the information of the first item is recorded in the temporary packing space. The length, width, and height information of the recorded items is put into the temporary bin record(l_t, w_t, h_t), update the temporary packing space of this bin using the Eq.(19).

$$V_t = V_{t-1} + v_i, quantity = quantity + 1, CL[quantity] = x_i \tag{19}$$

According to Eq.(19), the occupied volume of this bin is added to the current item, the number of packed items is increased by one, and the packed item is recorded using the list.

```
Algorithm 2: Packing method of the first item in a bin
Input: The set of items after reordering by position sequence$IRPS$, the length, width,
height
of$B_k$:$L_k, W_k, H_k$.
Output:$V, quantity, CL$.
Initialize: A temporary bin of $L_t = W_t = H_t = 0$ denoting the length, width, and
height
of the item already occupied by the bin. This bin is temporarily packed with a
set of
items,$CL = \varnothing$, the number of items$quantity = 0$. The length, width, and height
of the
temporary packing space of an item when it is not packed into a bin is$l_t = w_t =
h_t = 0$.
if $V_t = 0$ then
if $h_i > H_k$ or $w_i > W_k$ or $l_i > L_k$ then
```

(*continued on next page*)

(*continued*)

```
Algorithm 2: Packing method of the first item in a bin
Input: The set of items after reordering by position sequence$IRPS$, the length, width, height
of$B_k$:$L_k, W_k, H_k$.
Output:$V, quantity, CL$.
Initialize: A temporary bin of $L_t = W_t = H_t = 0$ denoting the length, width, and height
        break
    end if
    if $h_i < H_k$ or $w_i < W_k$ or $l_i < L_k$ then
        $l_t = l_i, w_t = w_i, h_t = h_i, L_t = L_t + l_i$
        update the temporary packing space of this bin by Eq.(19)
    end if
end if
```

Thirdly, Compare each packed item with the first, and judge the placement of the item and the place method if the packing conditions are satisfied. The euclidean distance determines the placement and orientation of the items. The euclidean distance is defined as$dist(x_i, x_1)$. The smaller the gap in length, width, and height between the item and the first item, the less lost space there will be because stacking items of the same size allows (l_t, w_t, h_t) to be more similar and even the same size as the first pack on this layer, thus reducing narrow areas between multiple items.

$$\min dist(x_i, x_1) = \sqrt{(x_i - x_1)^2} \tag{20}$$

According to Eq. (20), the euclidean distance between the first item to be packed and the first item on the current layer is calculated. The orientation with the smaller euclidean distance is the placement orientation. The exact implementation process of the LBPS is described in Algorithm 3.

```
Algorithm 3: Layered bin packing strategy(LBPS)
Input: The set of items after reordering by position sequence$IRPS$; the length, width,
height of$B_k$:$L_k, W_k, H_k$;$CL$ and $l_t = w_t = h_t$ which has been packed the first
item;$quantity = 1$.
Output:$\frac{V_t}{V_k}, quantity, CL$.
for each item $x_i \in I$ do
    The agent determines the orientation of the item by Eq.(20) and then turns the item
    according to Eq.(5).
    if $h_i > H_k - H_t$ then
        setting$l_t = w_t = h_t = L_t = W_t = H_t = 0$;
        break
    else
        if $w_i \leqslant W_k - W_t$ and $l_i \leqslant L_k - L_t$ and $h_i \leqslant H_k - H_t$ then
            $L_t = L_t + l_i$
            $l_t = max(l_t, l_i), w_t = max(w_t, w_i), h_t = max(h_t, h_i)$
            update the temporary packing space of this bin using Eq.(19)
        else if $w_i \leqslant W_k - W_t$ and $l_i > L_k - L_t$ and $h_i \leqslant H_k - H_t$ then
            $L_t = l_i, W_t = W_t + w_t$
            $l_t = l_i, w_t = w_i, h_t = max(h_t, h_i)$
            update the temporary packing space of this bin using Eq.(19)
        else
            $H_t = H_t + h_t, L_t = l_i, W_t = w_i$
            $l_t = l_i, w_t = w_i, h_t = h_i$
            update the temporary packing space of this bin using Eq.(19)
        end if
end for
```

Algorithm 3 is to obtain a proper placement of each item in a single container under the condition that the sequence of packing positions is determined.

4.5. Multi-vehicle cooperative bin packing strategy

In the actual logistics bin packing scenario, all the items need to be packed into multiple bins, the quantity of the item is relatively large, and the sizes and types are different. In order to extend the bin packing problem from a single-bin problem to a multi-bin problem, we propose the MVCBPS, which packs all items onto a varying number of vehicles. The detailed flow is described in Algorithm 4.

Table 3
Information on seven types of items.

item type	(l_i, w_i, h_i)	number of items
1	(4,4,3)	180
2	(6,4,3)	180
3	(6,3,3)	180
4	(6,3,2)	180
5	(4,3,2)	180
6	(4,3,3)	180
7	(4,2,2)	120

```
Algorithm 4:Multi-vehicle cooperative bin packing strategy(MVCBPS)
Input:$B_K, IRPS, IRP, P, PS$
Output:$PL, quantities, CLA, PLA$
Initialize: set of $N$ items$I$, a large enough training step$T$, combining the item set $I$
and the position sequence $PS$ to obtain the repacked items' set$IRPS.quantities = \varnothing$,
$CLA = \varnothing, PLA = \varnothing$
for $k \in \{1, 2, ..., T\}$ do
    $quantuty = 0, CL_k = \varnothing, V_t = 0, R_k = \frac{V_t}{V_k}, PL_k = \varnothing, RLS_k = \varnothing, PL = \varnothing$
    get the number of item $quantity_{now}$ that has already been packed from $quantities_1$ to
    $quantities_{k-1}$.
    if $quantity_{now} == N$ then
        break
    end if
    for$i \in \{quantity_{now} + 1, ..., N - 1\}$. do
        if $x_i$ can be packed into the $B_k$ through the LBPS do
            $V_{t+1} = V_t + v_i, quantity = quantity + 1, CL_q = x_i$
            $RLS_{k,q} = \frac{V_{t+1}}{V_k}$
        else
            break
        end if
    end for
    $quantities_k = quantity$
    $CLA_k = CL$
    for$x_j \in CL, j = 1, 2, ..., c$ do
        $PL_j = IRP_j$
    end for
    $PLA_k = PL$
end for
```

Algorithm 4 can determine the item packing probability, item numbers information of item numbers, and quantity under a specific location sequence. This algorithm is an intermediate method to take over the output sequence of the Seq2Seq policy network and the baseline-based policy gradient method.

5. Experiments

The length, width, and height of the bins and the items to be packed are defined as integers; the maximum value of all item sizes is less than or equal to 1/2 of the smallest bin. The shape of the bin is square; there are four types of bins with lengths of 12, 16, 20, and 24, respectively, denoted as Bin-12, Bin-16, Bin-20, and Bin-24. Packed items are classified into seven types with a total quantity of 1200. All the packed items are fixedly generated to avoid the problem of randomly generated data with excessive variability and unbalanced quantity, which leads to undesirable packing results. Details of the seven types of items are shown in Table 3.

The Adam method was used as an optimizer, and the learning rate was 0.01. The discount factor $\gamma = 0.99$ in formula (13) and the baseline value updates the parameter $\lambda = 0.02$ in formula (17). All DRL models are implemented in PyTorch, and all experiments were run on an Intel CPU 9600KF computer with an NVIDIA 1080Ti GPU.

Table 4
Comparison of the average space utilization of the eight models.

	Bin-12	Bin-16	Bin-20	Bin-24
RS	61.2 %± 0.1 %	55.0 %± 0.1 %	51.1 %± 0.1 %	53.4 %± 0.1 %
SA	60.0 %± 0.1 %	52.8 %± 0.1 %	47.5 %± 0.1 %	47.2 %± 0.2 %
ACO	80.9 %± 0.1 %	62.9 %± 0.1 %	61.3 %± 0.1 %	66.0 %± 0.1 %
GA	62.9 %± 0.1 %	55.6 %± 0.1 %	51.7 %± 0.1 %	54.4 %± 0.1 %
DE	84.9 %± 0.3 %	65.6 %± 0.4 %	64.3 %± 0.7 %	68.5 %± 0.4 %
Tubu	61.5 %± 0.6 %	54.7 %± 1.2 %	52.2 %± 0.2 %	52.1 %± 0.1 %
S2SDRL	91.2 %± 0.1 %	68.4 %± 0.1 %	70.1 %± 0.1 %	73.0 %± 0.1 %

Table 5
Time required to obtain smooth Loss comparison.

	Bin-12	Bin-16	Bin-20	Bin-24
RS	499.0	485.6	526.6	527.0
SA	327.2	303.9	303.1	328.2
ACO	5678.5	5550.4	5066.5	5690.3
GA	6366.0	5260.1	5676.8	6545.7
DE	2246.0	2249.0	2254.7	2254.2
Tabu	10297.3	7421.0	8308.8	8947.1
S2SDRL	94.8	98.0	70.1	96.5

5.1. Packing space utilization comparison analysis

To verify the advantages of the S2SDRL method in space utilization, we select seven heuristic algorithms to compare with the S2SDRL. The parameters of the seven heuristic algorithms are set as follows: There are no hyperparameter settings for the randomized search algorithm (RS); the SA sets the initial temperature as 10^{5}°C and the cooling factor as 0.98; the GA sets the crossover probability as 0.8, the variance probability as 0.3, and the population size as 20; the ACO sets the population size as 40 and the pheromone volatility factor as 0.8; the differential evolution algorithm(DE) sets the individuals as 20, the variation operator as 0.5, and the crossover operator as 0.1; The length of the tabu table and the length of the candidate table in the Tabu are both 100. These six heuristic algorithms and DRL approaches were tested in four scenarios of Bin-12, Bin-16, Bin-20, and Bin-24. The space utilization is shown in Table 4.

From Table 4, the S2SDRL has the highest space utilization in all four circumstances. DE has the highest average space utilization among the six heuristics, the space utilization under Bin-24 can reach 68.9 %, which is only 4.0 % lower than S2SDRL, but the convergence time of DE is 23 times higher than the S2SDRL. Compared with the heuristic algorithm, S2SDRL works better because DRL can use Bi-LSTM and Attention to memorize items information and give more weight to certain items to get a better ordering policy.

To enhance the overall performance of the S2SDRL in terms of running time, we chose six different heuristics and compared the time necessary to achieve a smooth Loss with S2SDRL. Table 5 shows the experimental results of the required time to acquire the smooth loss in the four types of scenarios.

According to Table 5, the S2SDRL approach has the shortest running time in all four scenarios. The SA takes the shortest time of every heuristic algorithm running on our device, as short as 303.1 s, but the S2SDRL takes no more than 98 s. The Seq2seq method totally masters the correlation between items during the initial sorting process and sorts according to this correlation. The first sequence obtained is very near to the optimal solution. RL evaluated the correlations of these items, showing which correlations were more favorable, and then the position sequences were adjusted slightly so that the initial value of Loss in Fig. 9 was relatively small and smooth.

According to Tables 4 and 5, the DE has the highest packing space utilization, the SA takes the least time while improving solution efficiency frequently takes more time, but the S2SDRL model not only takes less time but also has a higher packing space utilization than DE. Compared with other heuristic algorithms, the S2SDRL model does not have the influence of disturbance factors, does not fall into the local optimum and makes unsatisfactory results, and does not have a random variable that influences the model results, so it converges and makes great model results. Regarding model improvement, we can add constraints to the environment without updating the Seq2Seq or RL modules, making the environment closer to a real scenario, which is difficult to achieve with other heuristics. As a result, S2SDRL has a bigger advantage in the multi-vehicle cooperative situation.

To indicate the improved generalization performance of the S2SDRL model, we chose six heuristics for a comparative analysis of generalization ability with the S2SDRL model. Item packing tests were carried out under six different packing scenarios, Bin-25, Bin-26, Bin-27, Bin-28, Bin-29, and Bin-30, with the average space utilization results shown in Fig. 7.

We observe that, although the space utilization of S2SDRL is not the best in six different scenarios, it is greater than these six heuristics algorithms in Bin-25,26,27,30. Moreover, in these six scenarios, the average space utilization of S2SDRL is 59.6 %, which is higher than the other six algorithms, with the corresponding average values of Tabu, DE, GA, RS, SA, and ACO being 49.75 %, 57.9 %,48.72 %, 48.05 %, 41.02 %,

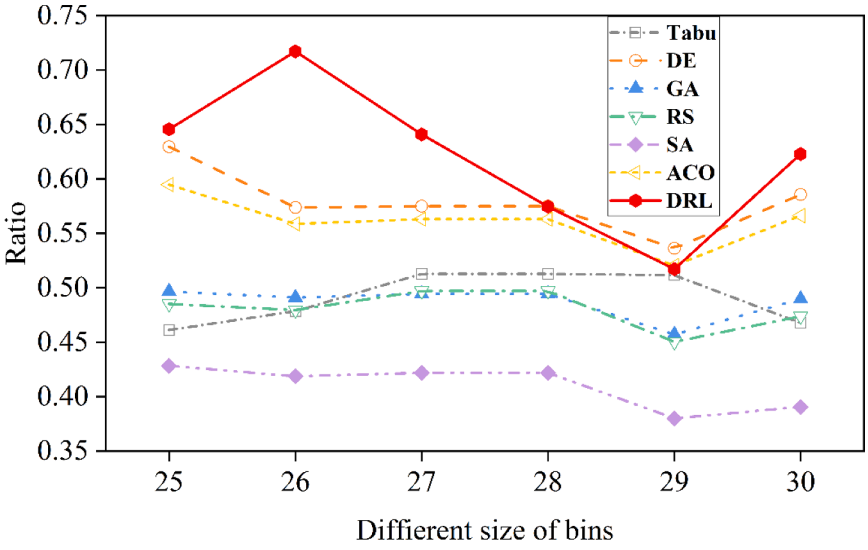

Fig. 7. Model generalization ability analysis of S2SDRL and some heuristic algorithms.

Table 6
Comparison of models' main modules.

	Bi-LSTM	activation function		Attention	Conv1D
		Sigmoid	Softplus		
Model 0(M0)	✓		✓	✓	
Model 1(M1)	✓	✓		✓	
Model 2(M2)	✓	✓		✓	✓
Model 3(M3)	✓		✓	✓	✓

and 56.10 %, respectively. In Bin-26, the space utilization of S2SDRL is 20 % lower than in Bin-29; the reason for the variation is the influence of LBPS. When there are fewer varieties of item types, the LBPS is more appropriate; when there are more types of items, the effect falls dramatically.

5.2. Ablation experiments

In order to further verify the effectiveness of each design module in the S2SDRL method, we designed four comparison models to confirm the efficacy of the selected models for the bin packing problem in the logistics scenario through experimental results. The main modules of the four models are shown in Table 6.

The four models were put in Bin-12, Bin-16, Bin-20, and Bin-24, respectively. The Baseline and Loss are depicted in Figs. 9 and 10, respectively.

From the average space utilization performance of the four models under four different packing scenarios in Fig. 8, Model 3 (M3) has the highest space utilization in all four scenarios. M0 is a model without adding a convolution module compared with M3, which has no multiple feature extraction for all item length, width, and height data. It is more difficult to find the correlation between items of different orders. M1 has the worst results of the four models because it does not use the convolution module and the softplus activation function, which shows the importance of the convolution module and the softplus activation function.

The loss fluctuation scope of Model 3 (M3) is minimal and relatively smooth in Fig. 9. Fig. 8 and Fig. 9 show the validity of the S2SDRL model as well as the effectiveness of each module for position sequence prediction. In the experiments, we found that the standard deviation of the model's loss becomes zero in some situations, but this does not indicate that there is no more adjustment between items' packing sequence because the loss will continue to change until it convergence.

5.3. Packing sequence and space utilization analysis

In the following, we pack the items into Bin-12, Bin-16, Bin-20, and Bin-24, respectively. The information of the items was packed into the bins, as shown in Fig. 10, where the horizontal axis represents the serial code of the bins, and the vertical axis represents the space utilization. The packing performance of the S2SDRL method is compared with the best performance and optimal packing case of the DE algorithm in four scenarios, respectively.

According to Fig. 10, after packing all items using the S2SDRL model, except the last bin, which has a considerable fluctuation in space utilization, the space utilization of all the other bins changes much less than the DE algorithm under the best and worst cases. Under the same LBPS, the overall bin space utilization of the heuristic method is arranged from small to large because the heuristic method packs most items of the small volume in the front bins first, resulting in lower space utilization in the front bins. The S2SDRL method does not place most of the small items in the position sequence early, so the overall packing results are

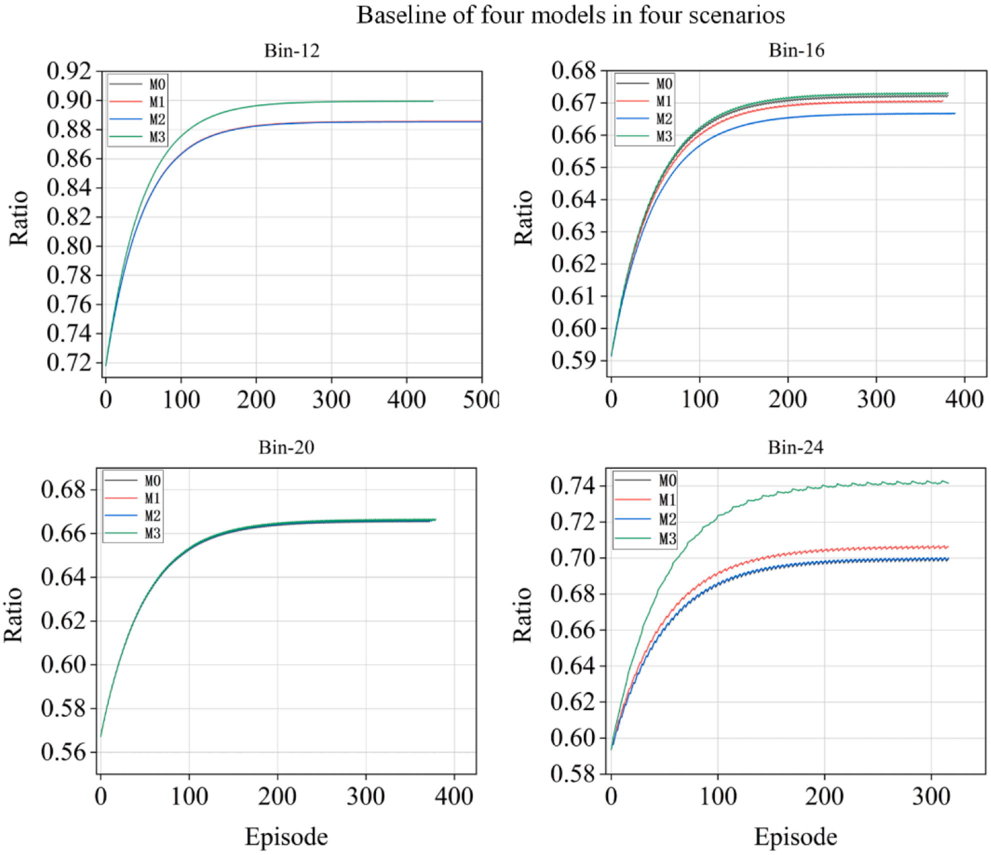

Fig. 8. The horizontal axis is the number of iterations, and the vertical axis is the average space utilization to represent the baseline value. Plots (a), (b), (c) and (d) represent the baseline at Bin-12, Bin16, Bin-20 and Bin-24 respectively. M3 is the S2SDRL model proposed in this article.

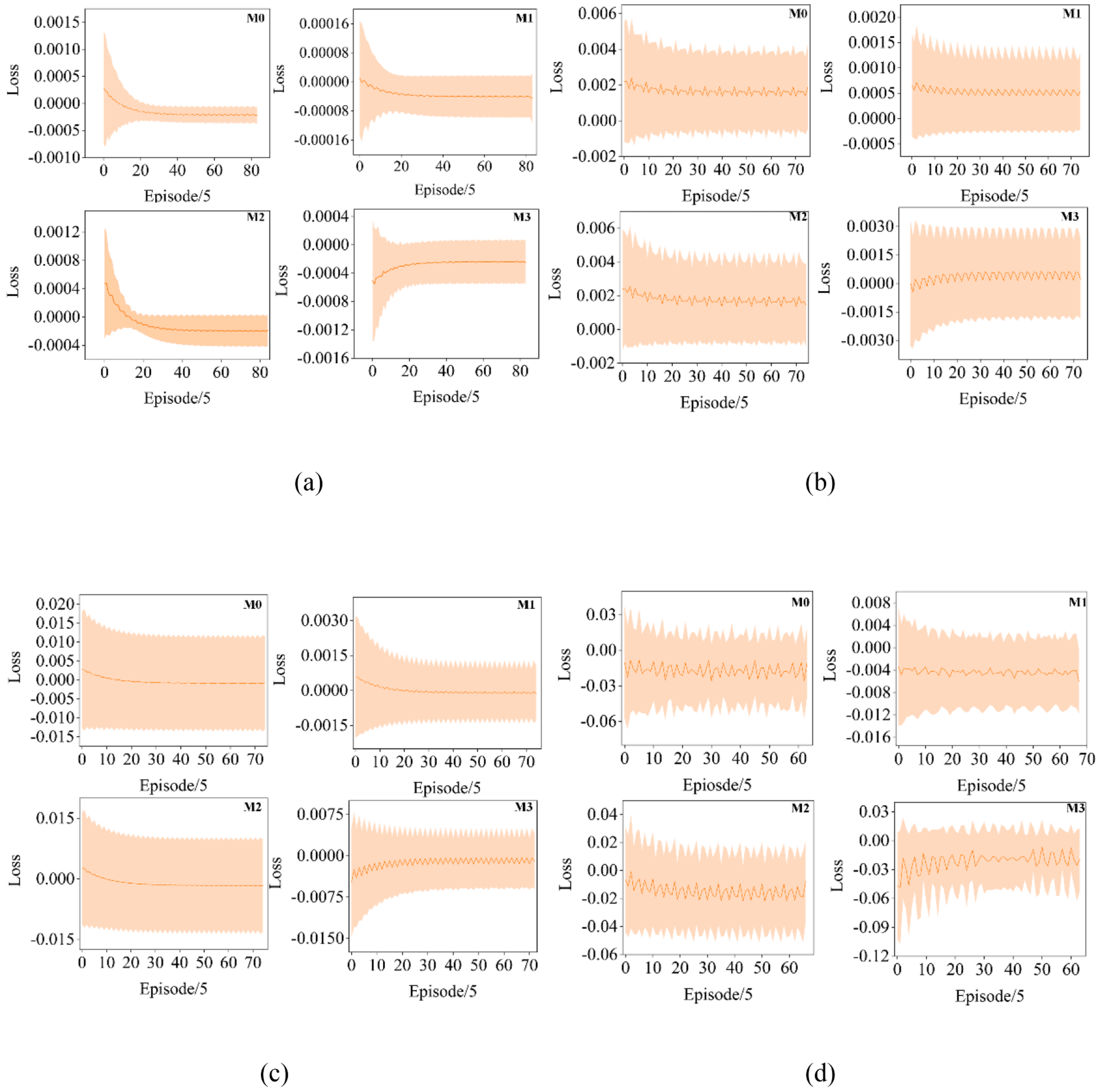

Fig. 9. The horizontal axis is the number of continuous iterations, and the vertical axis is the mean loss value. Plots (a), (b), (c), and (d) correspond to Bin-12, Bin16, Bin-20, and Bin-24 respectively.

better than the heuristic method. Therefore, the S2SDRL method requires less number of bins than the heuristic method in most scenarios.

5.4. *Sensitivity analysis*

This paper analyzes the sensitivity of the S2SDRL model using a univariate modification approach.

In addition to Table 3, five more group data are added to this study using item size as the sole influencing factor for the input data. The length, width and height of each group are shown in Table 7.

The exact number of item types as same as Table 3, with seven classes in each data set. The space utilization and the number of bins are shown in Table 8. We record the data in Table 3 as items_6.

As shown in Table 8, Bin-12 has about 20 % more space utilization than Bin-24. Items_1,items_2, items_3, items_4, items_5 and items_6 also correspond to Bin-24, which is about 20 % lower than Bin-12. It can be concluded that although some space utilization is just over 50 % when the item size is modified as a single variable, the space utilization is still stable concerning the number of bins. During the experiment, the longest running time was no more than 250 s, and the shortest was only 92 s.

This article uses the number of item classes as a single influencing factor for the input data. The items are divided into two classes, with the same size of items as the previous two classes of Table 3, and the total number of items is 1200, with 600 items in each class. The experimental results of Bin-12 and Bin-24 are respectively 88.89 %/12, 78.12 %/12, 80.00 %/6, 73.09 %/5, which displays the space utilization and the number of bins used, with the space utilization remaining high.

In this article, the total number of items as a single influence factor of the input data, the total number of items becomes 12,000, the size is the same as Table 3, the number of each class of items is increased by ten times, and the “space utilization/number of occupied boxes/time to find the best” after packing 12,000 items. The experimental results of Bin-12 and Bin-24 are respectively 92.81 %/315/1766.0, 71.29 %/173/1823.6, 73.43 %/86/1604.1, 89.13 %/41/1524.2. The number of bins increases by a multiple of 10 compared to Table 3, and the search time increases significantly, but the space utilization increases again. When experimenting with one scenario, the number of the bin is not unique,

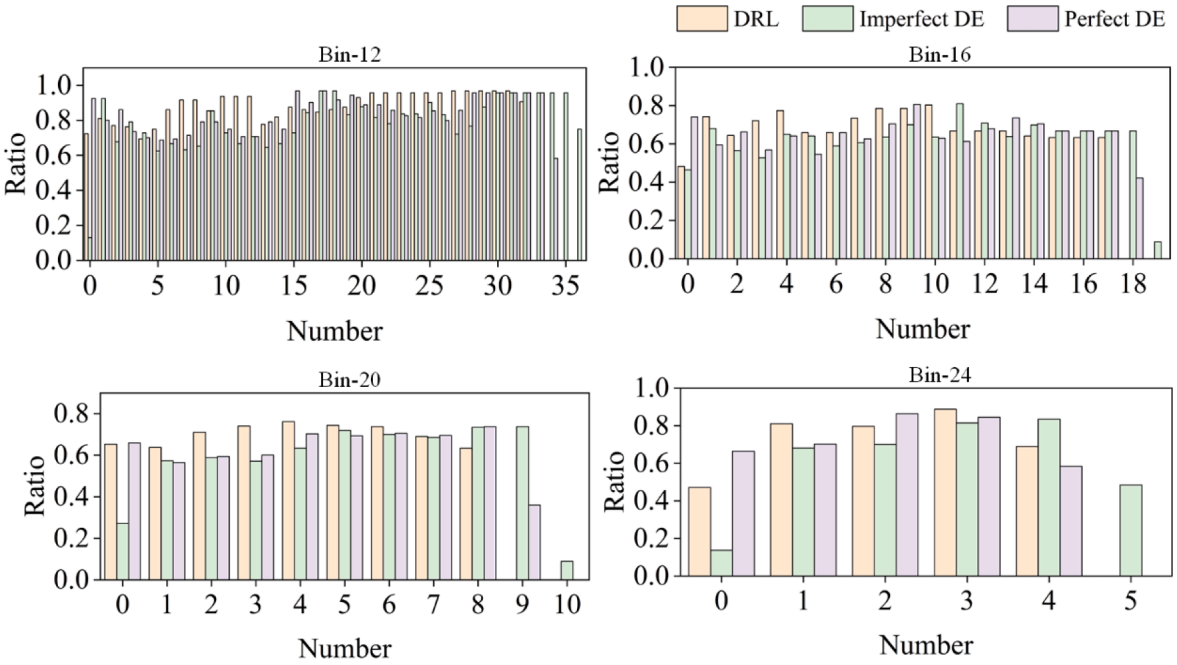

Fig. 10. The number of bins and the space utilization of each bin in Bin-12, 16, 20, and 24.

Table 7
Five different sets of item size data.

	items_1	items_2	items_3	items_4	items_5
1	(3,4,3)	(2,2,2)	(1,2,2)	(5,5,5)	(5,5,5)
2	(4,4,3)	(2,2,3)	(2,1,3)	(3,3,3)	(2,2,2)
3	(3,3,3)	(3,3,3)	(3,3,1)	(4,4,4)	(1,1,1)
4	(3,3,2)	(3,3,2)	(1,3,2)	(4,3,2)	(3,3,3)
5	(4,3,2)	(4,4,1)	(2,4,1)	(3,2,1)	(3,2,1)
6	(4,3,3)	(4,2,2)	(5,2,2)	(5,4,3)	(4,3,2)
7	(3,2,2)	(5,3,4)	(5,3,4)	(1,2,4)	(5,5,6)

Table 8
Comparison of six groups of items bin packed.

	Bin-12	Bin-16	Bin-20	Bin-24
items_1	85.42 %/24	72.00 %/12	73.64 %/6	64.08 %/4
items _2	79.15 %/18	75.22 %/8	61.61 %/5	57.43 %/3
items _3	74.25 %/13	67.87 %/6	67.98 %/3	55.54 %/2
items _4	75.24 %/43	72.01 %/19	77.83 %/9	57.91 %/7
items _5	72.00 %/42	67.11 %/19	72.58 %/9	47.23 %/8
items_6	91.20 %/33	68.40 %/18	70.10 %/9	73.00 %/5

but the difference between the number of the bin for multiple trials is not greater than one bin. The above analysis shows that S2SDRL does not experience a slippery decline in space utilization when the total number of items increases and the model's output has some stability.

From the experiments on item size, the number of item classes, and the total quantity of items, it can be concluded from the sensitivity analysis that the S2SDRL model is highly adaptable and some stable to changes in the input items data.

6. Conclusion

In this article, an S2SDRL model is proposed to solve the multi-vehicle cooperative bin packing problem. The Bi-LSTM-based Seq2Seq policy network is used to predict the packing position sequence of the items to obtain better position sequences. S2SDRL improves average space utilization by 4.0 % compared to the DE method. The ablation experiments of the encoder and decoder confirm that the Conv1D module and the Softplus activation function have better predictions of position sequences. The packing sequence and space utilization analysis show that packing larger volume items first improves overall space utilization. The sensitivity analysis also illustrates the some stability of the model. Despite these promising results, questions remain. In this article, The experimental data are only for seven different sizes of items, and the study should be repeated with data from the logistics site. Several questions still remain to be answered. In the future, we will also add more constraints to make the DRL model more adaptable to logistics bin packing scenarios. The issue of the balance of space utilization between different bins is an intriguing one that could be usefully explored in further research.

Bin packing maximizes carrying capacity, so proper bin packing planning is required. In this article, we look at the multi-vehicle cooperative parking problem from the sequence perspective, and the S2SDRL model can find the packing sequence that maximizes space usage. In the actual scenario of pharmaceutical logistics, whether the goods can be placed precisely and in order on the conveyor belt depends on how well the outgoing collectors arrange the things. Thus collector management needs to be reinforced, it depends on where the warehouse workers are and what job they need to do whether it is possible to manage the items out of the warehouse in sequence. So the management difficulty in performing an excellent job of loading is the management of people and the management of the inbound and outbound software systems. At the same time, using ML to provide managerial counsel is bound to introduce some dangers.

CRediT authorship contribution statement

Ran Tian: Conceptualization, Methodology, Supervision, Funding acquisition. **Chunming Kang:** Software, Writing – original draft, Visualization, Data curation. **Jiaming Bi:** Methodology, Software, Writing – original draft. **Zhongyu Ma:** Investigation, Writing – review & editing. **Yanxing Liu:** Supervision. **Saisai Yang:** Validation. **Fangfang Li:** Writing – original draft.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

This work is supported in part by the National Natural Science Foundation of China (71961028, 62261050), the Key Research and Development Program of Gansu (22YF7GA171), the Scientific Research Project of the Lanzhou Science and Technology Program (2018-01-58, 2017-4-101), and the Natural Science Foundations of Gansu (21JR7RA119, 21JR7RA208).

Data availability

Data will be made available on request.

References

Abeysooriya, R. P., Bennell, J. A., & Martinez-Sykora, A. (2017). Efficient local search heuristics for packing irregular shapes in two-dimensional heterogeneous bins[C]. In *International Conference on Computational Logistics* (pp. 557–571). Cham: Springer.

Baldi, M. M., Manerba, D., Perboli, G., et al. (2019). A generalized bin packing problem for parcel delivery in last-mile logistics[J]. *European Journal of Operational Research, 274*(3), 990–999.

Bian, Z., Shao, Q., & Jin, Z. (2016). Optimization on the container loading sequence based on hybrid dynamic programming[J]. *Transport, 31*(4), 440–449.

Crainic, T. G., Perboli, G., & Tadei, R. (2009). TS2PACK: A two-level tabu search for the three-dimensional bin packing problem[J]. *European Journal of Operational Research, 195*(3), 744–760.

Dell Amico, M., Furini, F., & Iori, M. (2020). A branch-and-price algorithm for the temporal bin packing problem[J]. *Computers & Operations Research, 114*, Article 104825.

Deng, W., Yao, R., Zhao, H., Yang, X., & Li, G. (2019). A novel intelligent diagnosis method using optimal LS-SVM with improved PSO algorithm[J]. *Soft Computing, 23* (7), 2445–2462.

Duan, L., Hu, H., Qian, Y., Gong, Y., Zhang, X., Wei, J., Xu, Y., 2018. A multi-task selected learning approach for solving 3D flexible bin packing problem[J]. arXiv preprint arXiv:1804.06896.

Elhedhli, S., Gzara, F., & Yildiz, B. (2019). Three-dimensional bin packing and mixed-case palletization[J]. *INFORMS Journal on Optimization, 1*(4), 323–352.

Fang, J., Rao, Y., Liu, P., et al. (2021). Sequence Transfer-Based Particle Swarm Optimization Algorithm for Irregular Packing Problems[J]. *IEEE Access, 9*, 131223–131235.

He, Y., & Wang, X. (2021). Group theory-based optimization algorithm for solving knapsack problems[J]. *Knowledge-Based Systems, 219*, Article 104445.

Hendalianpour, A. (2020). Optimal lot-size and price of perishable goods: A novel game-theoretic model using double interval grey numbers[J]. *Computers & Industrial Engineering, 149*, Article 106780.

Hendalianpour, A., Hamzehlou, M., Feylizadeh, M. R., et al. (2020). *Coordination and competition in two-echelon supply chain using grey revenue-sharing contracts[J]*. Grey Systems: Theory and Application.

Hu H, Zhang X, Yan X, et al. Solving a new 3d bin packing problem with deep reinforcement learning method[J]. arXiv preprint arXiv:1708.05930, 2017.

Hu, R., Xu, J., Chen, B., Gong, M., Zhang, H., & Huang, H. (2020). TAP-Net: Transport-and-pack using reinforcement learning[J]. *ACM Transactions on Graphics (TOG), 39* (6), 1–15.

Jiang, Y., Cao, Z., & Zhang, J. (2021). Learning to Solve 3-D Bin Packing Problem via Deep Reinforcement Learning and Constraint Programming[J]. *IEEE transactions on cybernetics, 1*, 1–13.

Kämpke, T. (1988). Simulated annealing: Use of a new tool in bin packing[J]. *Annals of Operations Research, 16*(1), 327–332.

Kasap, N., & Agarwal, A. (2012). Augmented neural networks and problem structure-based heuristics for the bin-packing problem[J]. *International Journal of Systems Science, 43*(8), 1412–1430.

Kosasih, E. E., & Brintrup, A. (2022). A machine learning approach for predicting hidden links in supply chain with graph neural networks[J]. *International Journal of Production Research, 60*(17), 5380–5393.

Kramer, R., Dell'Amico, M., & Iori, M. (2017). A batching-move iterated local search algorithm for the bin packing problem with generalized precedence constraints[J]. *International Journal of Production Research, 55*(21), 6288–6304.

Kucukyilmaz, T., & Kiziloz, H. E. (2018). Cooperative parallel grouping genetic algorithm for the one-dimensional bin packing problem[J]. *Computers & Industrial Engineering, 125*, 157–170.

Kundu, O., Dutta, S., & Deep-pack, K. S. (2019). In *A vision-based 2d online bin packing algorithm with deep reinforcement learning[C]//2019* (pp. 1–7).

Landero, V., Ríos, D., Pérez, J., Cruz, L., & Collazos-Morales, C. (2020). *Characterizing and analyzing the relation between bin-packing problem and tabu search algorithm[C]// International Conference on Computational Science and Its Applications* (pp. 149–164). Cham: Springer.

Levine, J., & Ducatelle, F. (2004). Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society, 55* (7), 705–716. https://doi.org/10.1057/palgrave.jors.2601771

Liang, H., Zou, J., Zuo, K., & Khan, M. J. (2020). An improved genetic algorithm optimization fuzzy controller applied to the wellhead back pressure control system [J]. *Mechanical Systems and Signal Processing, 142*, Article 106708.

Liu M, Man X, Zheng F, et al. An integer programming model for the single container loading problem with axle weight constraints[C]//2017 International Conference on Service Systems and Service Management. IEEE, 2017: 1-5.

Liu, P., & Hendalianpour, A. (2021). A branch & cut/metaheuristic optimization of financial supply chain based on input-output network flows: Investigating the Iranian orthopedic footwear[J]. *Journal of Intelligent & Fuzzy Systems*. Preprint): 1–19.

Liu, P., Hendalianpour, A., & Hamzehlou, M. (2021). Pricing model of two-echelon supply chain for substitutable products based on double-interval grey-numbers[J]. *Journal of Intelligent & Fuzzy Systems, 40*(5), 8939–8961.

Liu, P., Hendalianpour, A., Razmi, J., et al. (2021). A solution algorithm for integrated production-inventory-routing of perishable goods with transshipment and uncertain demand[J]. *Complex & Intelligent Systems, 7*(3), 1349–1365.

Martinovic, J., Strasdat, N., & Selch, M. (2021). Compact integer linear programming formulations for the temporal bin packing problem with fire-ups[J]. *Computers & Operations Research, 132*, Article 105288.

Nagar, D., Raghav, S., Bhardwaj, A., et al. (2021). *Machine learning: Best way to sustain the supply chain in the era of industry 4.0[J], 47*:, 3676–3682.

Nayal, K., Raut, R. D., Queiroz, M. M., et al. (2021). Are artificial intelligence and machine learning suitable to tackle the COVID-19 impacts? An agriculture supply chain perspective[J]. The. *International Journal of Logistics Management*.

Peters, F., & Schyns, M. (2020). Improving e-commerce logistics with Augmented Reality and Machine Learning: The case of the 3D bin packing problem[C]//6th International AR VR. *Conference.*

Saikia S, Verma R, Agarwal P, Shroff G, Vig L, & Srinivasan A. Evolutionary RL for Container Loading[J]. arXiv preprint arXiv:1805.06664, 2018.

Santos, L. F. O. M., Iwayama, R. S., Cavalcanti, L. B., Turi, L. M., de Souza Morais, F. E., Mormilho, G., et al. (2019). A variable neighborhood search algorithm for the bin packing problem with compatible categories[J]. *Expert Systems with Applications, 124*, 209–225.

Sardar, S. K., Sarkar, B., & Kim, B. (2021). Integrating machine learning, radio frequency identification, and consignment policy for reducing unreliability in smart supply chain management[J]. *Processes, 9*(2), 247.

Schroeder, M., & Lodemann, S. (2021). A systematic investigation of the integration of machine learning into supply chain risk management[J]. *Logistics, 5*(3), 62.

Tijjani S, Bucak İ Ö. An approach for maximizing container loading and minimizing the waste of space using Q-learning[C]//2013 The International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE). IEEE, 2013: 235-238.

Verma R, Singhal A, Khadilkar H, Basumatary A, Nayak S, Singh H V, Sinha R. A Generalized Reinforcement Learning Algorithm for Online 3D Bin-Packing[J]. arXiv preprint arXiv:2007.00463, 2020.

Yuan Y, Tole K, Ni F, et al. Adaptive Simulated Annealing with Greedy Search for the Circle Bin Packing Problem[J]. arXiv preprint arXiv:2108.03203, 2021.

Zazgornik, J., Gronalt, M., & Hirsch, P. (2012). The combined vehicle routing and foldable container scheduling problem: A model formulation and Tabu Search based solution approaches[J]. *INFOR: Information Systems and Operational Research, 50*(4), 147–162.

Zhang J, Zi B, Ge X. Attend2Pack: Bin Packing through Deep Reinforcement Learning with Attention[J]. arXiv preprint arXiv:2107.04333, 2021.

Zhao H, She Q, Zhu C, Yang Y, & Xu K. Online 3D Bin Packing with Constrained Deep Reinforcement Learning[J]. arXiv preprint arXiv:2006.14978, 2020.

Zhao H, Zhu C, Xu X, Huang H, & Xu K. Learning practically feasible policies for online 3D bin packing[J]. arXiv preprint arXiv:2108.13680, 2021.

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