Cavity-Aware Deep Reinforcement Learning for 3D Bin Packing

Wenjiao Xie[1], Sirui Wang[1], Yunqing Rao[1(✉)]and Qianhang Lyu[1]

[1] Huazhong University of Science and Technology, Wuhan, China
ryq@hust.edu.cn

Abstract. The three-dimensional bin packing problem (3D-BPP) is a classical optimization problem in logistics and transportation. Existing methods often rely on handcrafted heuristics or simplified state representations, which limits their ability to capture complex spatial relationships in packing environments. To address this issue, this paper proposes Cavity-Map-based Deep Reinforcement Learning (CMDRL) for the 3D-BPP. The proposed method introduces a cavity-map representation to model the geometric structure of free space and designs an enhanced spatiotemporal attention mechanism to jointly capture packing sequence dependencies and spatial layout information. A Transformer-based policy network trained with the Soft Actor-Critic (SAC) algorithm is developed to generate efficient packing decisions. Experimental results show that the proposed method outperforms several state-of-the-art DRL baselines in terms of gap ratio. Ablation studies further demonstrate the effectiveness of the cavity map and the spatiotemporal attention mechanism.

Keywords: 3D-BPP, deep reinforcement learning, cavity map, spatiotemporal attention mechanism.

1 Introduction

Deep reinforcement learning (DRL) has recently shown strong potential for solving sequential decision-making problems in combinatorial optimization, including the three-dimensional bin packing problem (3D-BPP) [1, 2]. As a classical NP-hard combinatorial optimization problem, the 3D-BPP aims to arrange items within a constrained space to maximize space utilization and improve transportation efficiency [3]. In this work, we specifically consider space utilization as the optimization objective. In our setting, the container's base dimensions (length and width) are fixed, while the height is unbounded. The packing process in 3D-BPP can be naturally formulated as a sequential decision-making problem, where the agent makes a sequence of packing decisions under the evolving container state to maximize space utilization. Compared with traditional heuristic approaches [4, 5], DRL can learn decision policies directly from data, reduce reliance on manually designed rules, and capture complex packing patterns through self-directed training. Representative DRL-based methods, such as TAP-Net and Attend2Pack, have demonstrated the feasibility of applying neural policy learning to packing optimization [6, 7].

However, existing DRL-based packing methods still exhibit important limitations. Many approaches represent the container state using simplified 2D top-down projections, which often overlook fine-grained three-dimensional contact relationships between packed items and container boundaries, leading to imprecise spatial modeling. In addition, existing methods often lack effective fusion of heterogeneous features and fail to capture the dynamic coupling between the temporal evolution of the packing sequence and the current spatial configuration of the container. These limitations may result in suboptimal packing decisions and reduced space utilization. To address these limitations, this paper proposes a Cavity-Map-based Deep Reinforcement Learning framework (CMDRL) for the 3D-BPP. The proposed framework combines a cavity-map representation with a Transformer-based policy network and is trained using the Soft Actor-Critic (SAC) algorithm. This design enables better modeling of the interaction between spatial layout and packing sequence. The main contributions of this paper are summarized as follows:

(1) A cavity-map representation for container state modeling is introduced using face-count and proximity indicators, enabling more effective characterization of spatial relationships among packed items and between items and the container.

(2) A spatiotemporal attention mechanism is designed to jointly capture the spatial structure of the container and the temporal dependencies in the packing sequence, improving feature interaction during decision making.

(3) A Transformer-based deep reinforcement learning framework trained with SAC is developed for effective policy learning in 3D-BPP.

2 Related Works

Existing studies on the 3D-BPP can be broadly divided into traditional optimization methods and learning-based approaches. Early work mainly focused on exact optimization and heuristic search. Padberg [8] proposed a mixed-integer programming formulation for three-dimensional packing, and Martello et al. [9] developed a branch-and-bound algorithm for feasible configuration search. To improve efficiency on larger instances, many heuristic and metaheuristic methods such as layer-building [10], genetic algorithms [11], and beam-search-based methods [12] were further introduced. Although these methods can achieve satisfactory performance in specific scenarios, they usually rely heavily on manually designed rules and have limited generalization ability.

With the development of machine learning, learning-based methods have gradually been applied to packing optimization. Hu et al. [13] introduced pointer networks to learn packing sequences for three-dimensional packing tasks. while Li et al. [14] proposed a conditional query learning model that incorporates historical actions into an attention-based architecture. More recent DRL-based approaches have further extended learning-based packing to sequential decision making. More recently, Jiang et al. [15] proposed a multimodal reinforcement learning framework that jointly optimizes item sequence, orientation, and placement decisions, and Que et al. [16] adopted Transformer-based policy and value networks to capture complex relationships among items

through self-attention. In addition, DRL has also been explored for online variants of the 3D-BPP. Zhao et al. [17, 18] formulated online packing as sequential decision-making and developed reinforcement-learning-based optimization frameworks.

Overall, existing DRL-based approaches have demonstrated strong potential for packing optimization. However, most existing DRL-based approaches still rely on simplified geometric representations of container states and lack effective mechanisms to capture the spatiotemporal dependencies in packing decisions. Unlike these methods, the proposed framework introduces a cavity-map representation and a spatiotemporal attention mechanism to better model the interaction between spatial layout and packing sequence.

3 Cavity-Map-Based Deep Reinforcement Learning Framework

This section presents the proposed CMDRL framework for 3D-BPP. In this framework, the packing process is formulated as a Markov decision process (MDP), where the box state is represented by a cavity map and the packing policy is learned through a Transformer-based architecture with spatiotemporal attention. The framework is trained using an SAC-style actor–critic strategy.

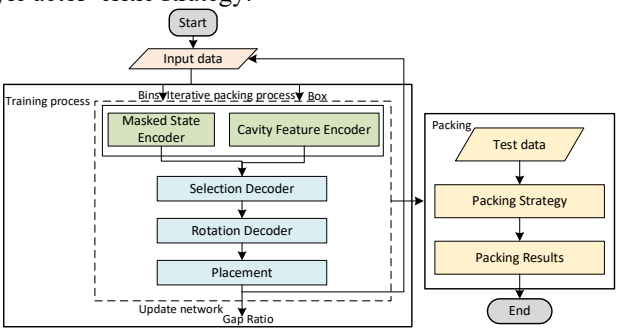


Fig. 1. Overall workflow of CMDRL.

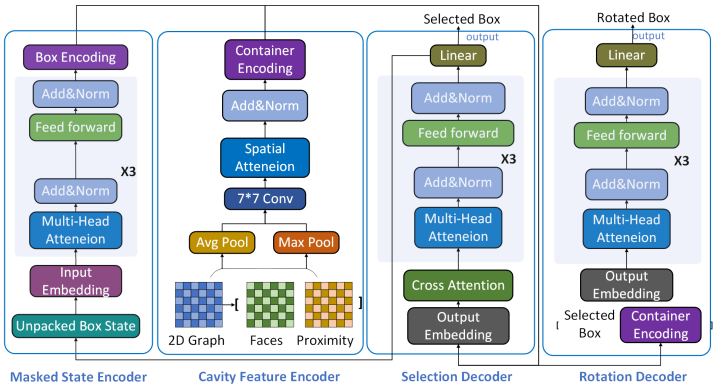


Fig. 2. Module structure of the packing process.

3.1 Framework Overview

The proposed CMDRL framework explicitly models the box state by introducing a cavity-map-based spatial representation. At each decision step, the current packing state and the remaining free space are encoded by a masked state encoder and a cavity feature encoder, respectively. The encoded features are then fed into the subsequent selection and rotation modules to generate packing decisions. The overall workflow of the framework is shown in Fig. 1, while the detailed module structure is illustrated in Fig. 2.

3.2 Problem Formulation

Packing Objective and Constraints. The box has fixed base dimensions *L* and *W*, while its height is determined dynamically during the packing process. Given a set of *N* rectangular bins $\{(l_i, w_i, h_i)\}_{i=1}^{N}$, where l_i, w_i,and h_i denote the length, width, and height of the $i_$th bin, respectively, the objective of the 3D-BPP is to place all bins into the box such that the final packing height is minimized, thereby maximizing overall space utilization. Let $(0,0,0)$ denote the lower-left corner of the box and (x_i, y_i, z_i) denote the coordinates of the lower-left corner of bin i. The packing efficiency is measured by the space utilization ratio $\eta = \frac{\sum_{(i=1)}^{N} w_i l_i h_i}{WLH_N}$ where H_N denotes the final box height after packing all bins. Maximizing η is equivalent to minimizing the final packing height.

To ensure feasible packing configurations, several constraints must be satisfied. First, bins must be placed orthogonally such that their faces remain parallel to the box surfaces. Second, boundary constraints ensure that bins do not exceed the box limits. Third, overlap constraints guarantee that bins do not intersect with each other. In addition, each bin can be rotated by $90°$ along the x, y, and z axes, resulting in six possible orientation configurations. Finally, mechanical stability constraints are considered to ensure that the packing configuration is physically feasible. Specifically, the center of gravity of a bin must lie within the support region formed by the bins underneath it. A placement is considered stable if the centroid of the bin is fully supported by an existing bin or lies within the convex hull formed by multiple supporting bins.

MDP Formulation. To solve the 3D-BPP within a DRL framework, the packing process is formulated as an MDP. At each packing step, the proposed framework performs two learned decisions: bin selection and rotation selection. Specifically, the policy first selects an unpacked bin from the remaining set, and then determines its orientation among six feasible rotation configurations. Given the selected bin and its orientation, the final placement position is determined by an EMS-based left-bottom heuristic.

Directly modeling the joint decision process would lead to a large combinatorial action space. Therefore, the policy is factorized using the chain rule as:

$$\begin{aligned} P_t(a_t \mid s_t) &= P_t(as_t, ao_t, ap_t \mid s_t) \\ &= P_t(ap_t \mid s_t)\, P_t(ao_t \mid ap_t, s_t)\, P_t(as_t \mid ao_t, ap_t, s_t) \end{aligned} \tag{1}$$

Among them, as_t represents the probability of selecting i_th bin under the current state s_t, ao_t represents the probability of rotating the selected i_th bin in a certain direction.

3.3 Cavity Map Representation

Cavity Feature Encoding. To characterize the spatial structure of free space, two cavity-related descriptors are introduced, namely the number of faces M_{faces} and the proximity M_{prox}.The face-count feature is defined as:

$$M_{faces}(x,y) = 1\big(hm_{diffx}(x,y) > 0\big) + 1\big(hm_{diffy}(x,y) < 0\big) \tag{2}$$

where M_{faces} indicates whether the current position is in contact with other objects along the x- and y-directions. If the difference in a certain direction is greater than zero, it implies that there exists an object in contact in that direction. Therefore, M_{faces} reflects the local support strength and spatial compactness of the packing configuration. In general, a larger value of M_{faces} implies stronger static stability and a more compact arrangement.

To further model the global spatial distribution of free space, a proximity feature is introduced. The distance from a candidate position (x, y) to the center of the box is calculated as:

$$d_i = \sqrt{(x - L/2)^2 + (y - W/2)^2} \tag{3}$$

Based on this distance, the proximity map is defined as:

$$M_{prox}(x,y) = \exp\left(-d_i / \sqrt{(L/2)^2 + (W/2)^2}\right) \tag{4}$$

where L and W denote the length and width of the box, respectively. A larger value of M_{prox} indicates that the position is closer to the box center. Since the placement strategy in this work tends to prioritize bottom-corner spaces, the proximity feature helps the model reduce fragmented edge space and improve packing compactness.

Spatial Attention-Based Cavity Map. After computing the cavity-related features, the spatial feature map is represented as $M_t \in R^{4\times H\times W}$.To further highlight informative regions in the feature map, a spatial attention mechanism is introduced to generate the cavity representation h_t:

$$\begin{aligned} h_t &= \sigma\big(f^{7\times 7}([AvgPool(M_t); MaxPool(M_t)])\big) \\ &= \sigma\big(f^{7\times 7}\big[F_{avg}^{M_t}; F_{max}^{M_t}\big]\big) \end{aligned} \tag{5}$$

where $F_{avg}^{M_t}$ and $F_{max}^{M_t}$ denote the global average pooling and global maximum pooling results, respectively. A 7×7 convolution is then applied to generate a spatial attention

weight matrix, followed by a sigmoid activation function. The resulting h_t serves as the output of the cavity feature encoder.

Through the above design, the cavity map explicitly integrates both local support information and global spatial distribution, enabling the model to better identify feasible and compact placement regions. Compared with conventional 2D state representations, the proposed cavity-map representation provides a more informative description of the 3D packing environment and forms the basis for the subsequent spatiotemporal decision process.

3.4 Spatiotemporal Decision Modules

Based on the cavity-map-based spatial representation introduced in the previous subsection, the proposed framework further incorporates a spatiotemporal attention mechanism to jointly model the temporal packing sequence and the spatial structure of the box. The purpose of this mechanism is to enable the agent to determine not only which bin should be selected next, but also how it should be oriented and placed under the current spatial configuration. As shown in Fig. 2, we incorporate an enhanced spatiotemporal attention mechanism into the selection decoder and the rotation decoder, which together generate the packing action at each step.

Masked State Encoder. The masked state encoder tracks packing progress through a dynamic mask and encodes the remaining unpacked bins at each decision step. Let $B_t = \{b_t^1, b_t^2, \dots, b_t^N\}$ denote the bin set at step t, where $b_t^i = (l_t^i, w_t^i, h_t^i, x_t^i, y_t^i, z_t^i)$ represents the geometric attributes of bin i. A binary mask $m_t = \{m_t^1, m_t^2, \dots, m_t^N\}$ is used to indicate whether a bin has already been packed, where $m_t^i = 1$ means bin $i$has been packed and $m_t^i = 0$ otherwise. Based on the current mask, the input is updated to a masked bin set $B_t^{m_t}$, so that only valid candidate bins are encoded. The masked input is first projected through a linear layer:

$$\overline{X_t} = \mathrm{Linear}\left(B_t^{m_t}\right) \tag{6}$$

Then, stacked multi-head attention layers are applied to capture dependencies among the remaining candidate bins:

$$X_t = \mathrm{LayerNorm}\left(\overline{X_t} + \mathrm{MultiHead}(\overline{X_t})\right) \tag{7}$$

The output $X_t$serves as the temporal embedding of the remaining bin set and is fed into the subsequent selection and rotation modules.

Selection Decoder. The selection decoder serves as the core decision-making module of the proposed framework. Its goal is to select, at each time step, the unplaced bin that best matches the current remaining space. To achieve this, the decoder takes as input both the temporal embedding X_t produced by the masked state encoder and the cavity representation h_t generated by the cavity feature encoder. In this way, the decoder can

explicitly model the interaction between the packing sequence and the evolving spatial layout of the box.

To fuse the two modalities, a cross-attention module is first introduced, where the query is the box embedding X_t, and both the key and value are the cavity map h_t. The corresponding cross-attention output is computed as:

$$\tilde{e}_t = CrossAttention(X_t, h_t, h_t) \tag{8}$$

and the fused query vector is defined as:

$$q_t^s = Mean(\tilde{e}_t, h_t)^T \tag{9}$$

This operation dynamically aggregates spatial and temporal information, enabling the model to capture the correlation between the geometric properties of candidate bins and the available cavity regions in the box.

To further enhance the feature representation, a multi-head attention mechanism is applied. Specifically, the query q_t^s, key X_t, and value X_t are projected into multiple subspaces and processed in parallel:

$$\widetilde{q_t^s} = Concat(heads_1, \dots, heads_M)W^o \tag{10}$$

where each attention head is defined as:

$$heads_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{11}$$

Each attention head captures spatiotemporal correlation patterns at a different granularity, and W^o enotes the output projection matrix.

After obtaining the fused feature representation, an additional attention layer with a single head ($M = 1$) is used to compute the compatibility score for each candidate bin. A greedy strategy is then adopted to select the bin with the highest probability:

$$indext = argmaxp_{t,s} \tag{12}$$

$$p_{t,s,j} = softmax\left(C \cdot \tanh\left(W^Q e_t^{qs}\right)^T \left((W^K X_t^j)/\sqrt{h}\right) if\ m_t^j = True\right) \tag{13}$$

Here, m_t^jindicates whether the j_th block has already been packed into the box. If the block has already been placed, the corresponding mask is set to 1; otherwise, it is 0. This masking mechanism ensures that only valid candidate bins are considered during decoding.

Rotation Decoder. After the target bin is selected, the rotation decoder is responsible for determining its orientation in the box. Since a bin can be rotated by 90° along the x-, y-, and z-axes, the decoder outputs a probability distribution over six possible orientation configurations.

To fully describe the current packing context, the input of the rotation decoder consists of two parts: the embedding of the selected bin at time step t, denoted by $X_t^{Selected}$, and the cavity representation h_t.Their fused representation is computed as:

$$o_t = Mean(X_t^{Selected}, h_t) \tag{14}$$

Using the same attention mechanism as in the decoder, the glimpse vector is obtained as:

$$H_t = C \cdot \tanh(W^H o_t) \tag{15}$$

The glimpse vector is then mapped to a six-dimensional probability distribution through a fully connected layer:

$$p_{t,o} = softmax(H_t) \tag{16}$$

The rotation direction with the highest probability is selected as the final orientation decision. In this way, the model can adaptively determine the most suitable orientation according to both the bin geometry and the current cavity structure.

3.5 Placement Strategy.

Once the bin and its orientation are determined, the placement strategy executes the final insertion into the box. In this work, we adopt an LB placement strategy based on the maximal remaining-space EMS (Empty Maximal Space) representation [19]. For each EMS, the algorithm attempts to place the selected bin at its left-front-bottom corner and chooses the position that yields the highest packing reward.

To ensure physical feasibility, the placement must also satisfy a stability constraint. Specifically, the center of gravity of the selected bin is computed and checked against the support region $c_i \in I_{support}$. If the stability constraint is not satisfied, the current placement candidate is discarded, and the algorithm reselects the next best placement position from the remaining EMS list according to the original LB heuristic order. This design ensures that the generated packing solution is not only compact but also physically stable.

Through the above process, the proposed spatiotemporal attention mechanism enables the framework to effectively couple bin selection, rotation decision, and spatial placement under the evolving box state, thereby improving packing efficiency and decision quality.

3.6 SAC Training

The proposed framework is trained under the SAC paradigm. In the training process, The actor consists of the selection decoder and rotation decoder, while placement is executed by an EMS-based heuristic conditioned on the learned decisions. The critic follows a similar overall architecture to the actor. After capturing the spatiotemporal correlation through multi-head attention, the final glimpse query vector is obtained and fed into a multi-layer perceptron (MLP) to produce the state value scalar $V(s_t)$.

During training, we adopt the generalized advantage estimation (GAE) method to balance the variance and bias of the estimate, thereby improving the accuracy of the advantage estimation:

$$\hat{A}_t^{\gamma,\lambda} = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t-1}\delta_{T-1} \tag{17}$$

The temporal difference error $\delta_t = reward_t + \gamma V(s_{t+1}) - V(s_t)$, $\gamma \in [0,1]$ is the discount factor, $\lambda \in [0,1]$ is the bias–variance trade-off parameter, and T denotes the trajectory length.

To encourage compact packing layouts, this paper adopts the reward function proposed in George and Robinson [20]. The reward is defined based on the change in the current volume gap of the remaining space:

$$r_t = \frac{WLH_t - \sum_{i=1}^{t} w_i\, l_i h_i}{WLH_t + \epsilon} \tag{18}$$

$$reward_t = r_{t-1} - r_t \tag{19}$$

where H_t represents the height of the smallest bounding bin containing t bounding bins, and W and L denote the width and length of the box, respectively. ϵ is a very small positive real number. Since the reward directly reflects the change in the gap ratio, the agent is encouraged to improve space utilization during training.

The network is optimized using a composite loss function consisting of the actor loss, critic loss, and entropy loss:

$$L(\theta \mid s_t) = L(\theta_a \mid s_t) + L(\theta_c \mid s_t) + L\left(\theta_{entropy} \mid s_t\right) \tag{20}$$

$$L(\theta_a \mid s_t) = E_{a_t - p_{\theta_a}}\left[-\hat{A}_t^{\gamma,\lambda}(s_t, a_t) log p_{\theta_a}(a_t \mid s_t)\right] \tag{21}$$

$$L(\theta_c \mid s_t) = MSE\left(V(s_t), \left(\hat{A}_t^{\gamma,\lambda}(s_t, a_t) + V(s_t)\right)\right) \tag{22}$$

$$L\left(\theta_{entropy} \mid s_t\right) = -\alpha E_{a_t - p_\theta}[\bar{H} + log p_\theta(a_t \mid s_t)] \tag{23}$$

The actor loss encourages the policy to select actions with higher expected return, the critic loss improves the accuracy of value estimation, and the entropy loss promotes exploration and avoids premature convergence. Here, $log p_{\theta actor}(a_t \mid s_t)$ covers both the selection strategy and the rotation strategy, $\bar{H}$ is the target entropy, and α is an adjustable entropy coefficient. After computing the loss, the Adam optimizer is used to update the actor parameters θ_a and critic parameters θ_c.

4 Experiments

In this section, we conduct experiments and ablation studies to evaluate the effectiveness of the proposed method. We further analyze its training behavior, hyperparameter sensitivity, and scalability across different problem sizes.

4.1 Experimental Setup

Parameter Configuration. To ensure training stability and fair comparison, the parameter settings follow established baselines [21] while being adjusted to the proposed framework. The model is implemented in PyTorch and optimized using Adam [22]. The learning rates of the critic and actor are set to $1 \times 10^{-4}$and 1×10^{-5}, respectively, with a decay factor of 0.96 applied every 2000 steps. In the reinforcement learning setting, the discount factor is $\gamma = 0.99$, the GAE parameter is $\lambda = 0.98$, the reward scaling factor is set to $C = 10$, and the target entropy is fixed at 0.6. Gradient clipping with a threshold of 5 is adopted to improve training stability. The framework is trained for 100,000 steps on an NVIDIA RTX 4090 GPU.

Table 1. Comparison results of different algorithms on different examples.

Instance	GA (%)	CQL (%)	MM (%)	ATTEN2PACK (%)	DMRL (%)	CMDRL (%)
Bin10	27.2	27.4	-	-	27.0	24.28±0.0039
Bin16	31.5	30.6	-	-	22.8	22.69±0.0024
Bin20	33.9	32.4	28.2	23.3	21.1	18.33±0.0028
Bin30	37.4	30.2	24.5	20.3	18.9	17.75±0.0015

Dataset, Baselines, and Metrics. The dataset setting follows Li et al. [23]. The box size is fixed to 100×100, and the side lengths of the bins are uniformly sampled from integers between 20 and 80, yielding instances of varying packing difficulty. We evaluate the proposed method on Bin10, Bin16, Bin20, and Bin30, where the number denotes the number of bins in each instance.

To evaluate the effectiveness of the proposed method, we compare CMDRL with several representative algorithms reported in the literature, including genetic algorithm (GA) [24], conditional query learning (CQL) [23], multimodal deep reinforcement learning (MM) [15], Attend2Pack [6], and dynamic multimodal deep reinforcement learning (DMRL) [21]. The main evaluation metric is the gap ratio, where a lower value indicates better space utilization.

4.2 Experimental Results and Analysis

Comparison Results. Table 1 reports the comparison results on Bin10, Bin16, Bin20, and Bin30. CMDRL achieves the lowest gap ratios on Bin10, Bin20, and Bin30, and remains competitive on Bin16. Compared with the strongest DRL baseline considered in this study, DMRL, the proposed method reduces the gap ratio by 2.7%, 2.8%, and 1.2% on Bin10, Bin20, and Bin30, respectively. These results indicate that the cavity-map-based state representation and the spatiotemporal attention mechanism improve packing compactness and overall space utilization.

Training and Visualization Results. The training curves on different datasets are shown in Fig. 3(a). CMDRL exhibits a clear convergence trend and gradually stabilizes after approximately 90,000 iterations. As shown in Fig. 3(b)–(e), the average reward

increases steadily during training, while the actor loss, critic loss, and entropy loss all converge to stable ranges. These results suggest that the proposed framework can be optimized effectively and maintains stable training dynamics.

To further verify that the observed improvements are not due to random variation, paired t-tests are conducted between CMDRL and DMRL over repeated runs on the same benchmark instances. The improvements on Bin20 and Bin30 are significant at the 99% confidence level ($p < 0.01$), while the improvement on Bin10 is significant at the 95% confidence level ($p < 0.05$). These results support the effectiveness of the proposed method.

To qualitatively illustrate the packing effect, Fig. 4 shows the packing visualization results on Bin10, Bin16, Bin20, and Bin30. The proposed method can generate compact and stable packing layouts across different problem scales.

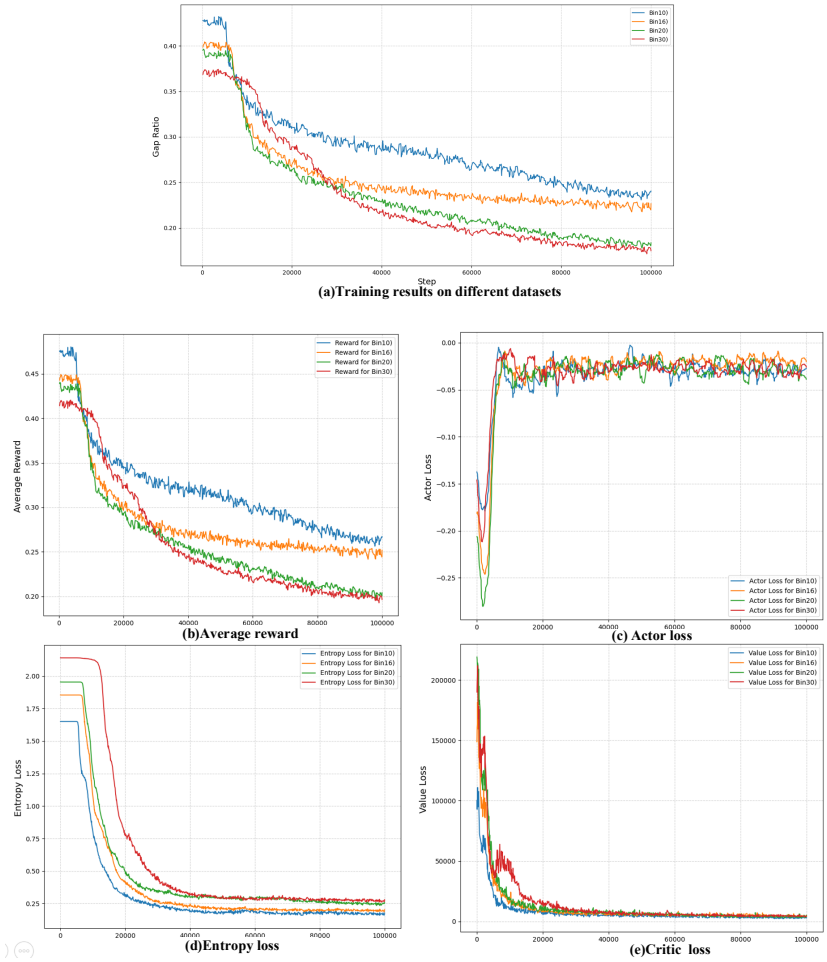


Fig. 3. Training performance of CMDRL on different datasets.

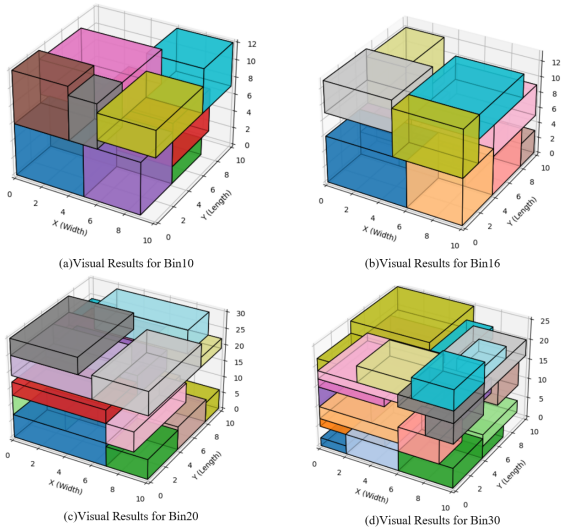


Fig. 4. Training visualization results of different datasets.

Ablation and Sensitivity Analysis. Ablation experiments are conducted on Bin20 to evaluate the contribution of each module. As shown in Fig. 5, removing either the cavity-map representation or the spatiotemporal attention mechanism leads to a higher gap ratio, indicating that both components contribute to the final performance. When the two modules are used together, the best result is obtained, with the gap ratio reduced by 2.8% relative to the baseline setting.

In the early stage of training, the performance gain is less pronounced when the two modules are combined. This may be due to different optimization preferences of the two modules during early learning. As training proceeds, their complementary effects become more evident, leading to improved packing compactness in later stages.

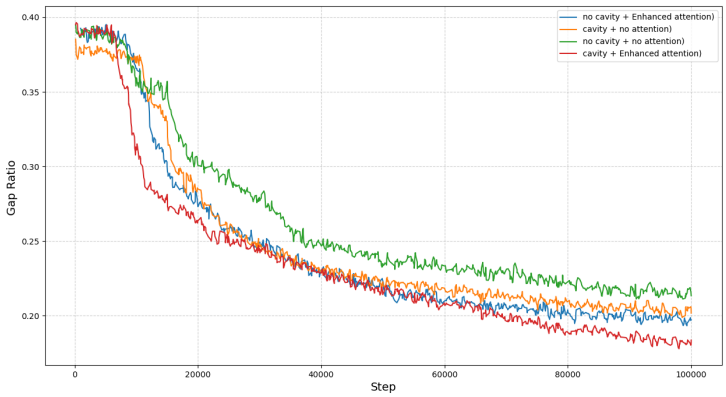
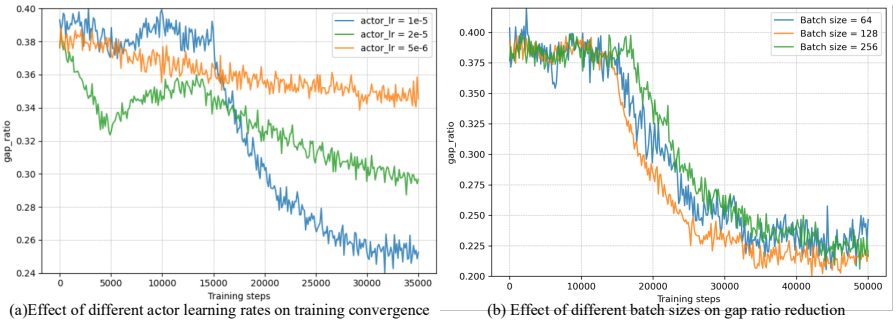


Fig. 5. Ablation experiment results of Bin20 dataset.

Sensitivity analyses on the actor learning rate and batch size are conducted on Bin20, as shown in Fig. 6. Among the tested learning rates, 1×10^{-5} achieves the best final gap ratio, while smaller values converge more slowly and larger values lead to unstable training. For batch size, 128 provides the best trade-off between convergence speed and final performance, whereas 64 introduces larger oscillations and 256 slows down convergence.



(a)Effect of different actor learning rates on training convergence (b) Effect of different batch sizes on gap ratio reduction

Fig. 6. Sensitivity analysis of hyperparameters.

Scalability Analysis. To evaluate scalability on larger instances, additional experiments are conducted on Bin50 and Bin100. As reported in Table 2, the total inference time increases from 1.36 s on Bin30 to 5.20 s on Bin100, showing a moderate growth trend with problem size. Meanwhile, the gap ratio remains stable and even slightly decreases from 17.75% to 16.94%. These results indicate that CMDRL maintains strong packing performance as the problem scale increases.

This scalability is also supported by the framework design. Traditional voxel-based methods require $O(L \times W \times H)$ spatial complexity, which becomes expensive for large boxes. In contrast, the proposed cavity-map representation projects the 3D state into 2D feature maps, reducing the spatial complexity to $O(L \times W \times C)$, where C is the number of feature channels. In addition, although self-attention has quadratic complexity with respect to the number of bins, the dynamic masking mechanism restricts computation to the unpacked bins at each step, reducing the effective computational cost as packing proceeds.

Table 2. Performance analysis for different problem scales.

Instance Size	Total inference time (s)	Gap ratio (%)	Feasibility
Bin30	1.36	17.75	Real-time
Bin50	2.40	17.21	Real-time
Bin100	5.20	16.94	Near real-time

5 Conclusion

This paper proposes CMDRL, a deep-reinforcement-learning-based framework for the 3D-BPP that integrates a cavity-map-based spatial representation with an enhanced spatiotemporal attention mechanism, to better capture container geometry and the dynamic interaction between packing sequence and spatial layout. By integrating these two components, the framework can learn more compact and stable packing policies with reduced reliance on handcrafted rules. Experimental results on multiple benchmark datasets demonstrate that the proposed method achieves improved space utilization and lower gap ratios compared with several state-of-the-art DRL baselines, which verifies the effectiveness of the proposed framework.

Future work will focus on improving the cavity-map representation for large-scale problems and further investigating the spatiotemporal attention mechanism to improve robustness and real-time decision capability.

Acknowledgments. This work is supported by National Natural Science Foundation of China under Grant 52475518.

References

1. Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S.: Neural Combinatorial Optimization with Reinforcement Learning, http://arxiv.org/abs/1611.09940, (2017).
2. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to Sequence Learning with Neural Networks, http://arxiv.org/abs/1409.3215, (2014).
3. Zhao, X., Bennell, J.A., Bektaş, T., Dowsland, K.: A comparative review of 3D container loading algorithms. Int. Trans. Oper. Res. 23, 287–320 (2016).
4. Gehring, H., ortfeldt, A.: A genetic algorithm for solving the container loading problem. Int. Trans. Oper. Res. 4, 401–418 (1997).
5. Li, X., Zhang, K.: A hybrid differential evolution algorithm for multiple container loading problem with heterogeneous containers. Comput. Ind. Eng. 90, 305–313 (2015).
6. Zhang, J., Zi, B., Ge, X.: Attend2Pack: Bin Packing through Deep Reinforcement Learning with Attention, http://arxiv.org/abs/2107.04333, (2021).
7. Hu, R., Xu, J., Chen, B., Gong, M., Zhang, H., Huang, H.: TAP-Net: transport-and-pack using reinforcement learning. ACM Trans Graph. 39, 232:1-232:15 (2020).
8. Padberg, M.: Packing small boxes into a big box. Math. Methods Oper. Res. 52, 1–21 (2000).
9. Martello, S., Pisinger, D., Vigo, D.: The Three-Dimensional Bin Packing Problem. Oper. Res. 48, 256–267 (2000).
10. Harrath, Y.: A three-stage layer-based heuristic to solve the 3D bin-packing problem under balancing constraint. J. King Saud Univ. - Comput. Inf. Sci. 34, 6425–6431 (2022).
11. Kang, K., Moon, I., Wang, H.: A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem. Appl. Math. Comput. 219, 1287–1299 (2012).
12. Araya, I., Moyano, M., Sanchez, C.: A beam search algorithm for the biobjective container loading problem. Eur. J. Oper. Res. 286, 417–431 (2020).

13. Hu, H., Zhang, X., Yan, X., Wang, L., Xu, Y.: Solving a New 3D Bin Packing Problem with Deep Reinforcement Learning Method, http://arxiv.org/abs/1708.05930, (2017).
14. Li, D., Gu, Z., Wang, Y., Ren, C., Lau, F.C.M.: One model packs thousands of items with Recurrent Conditional Query Learning. Knowl.-Based Syst. 235, 107683 (2022).
15. Jiang, Y., Cao, Z., Zhang, J.: Solving 3D Bin Packing Problem via Multimodal Deep Reinforcement Learning. In: Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems. pp. 1548–1550. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2021).
16. Que, Q., Yang, F., Zhang, D.: Solving 3D packing problem using Transformer network and reinforcement learning. Expert Syst. Appl. 214, 119153 (2023).
17. Zhao, H., She, Q., Zhu, C., Yang, Y., Xu, K.: Online 3D Bin Packing with Constrained Deep Reinforcement Learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 741–749 (2021).
18. Zhao, H., Zhu, C., Xu, X., Huang, H., Xu, K.: Learning practically feasible policies for online 3D bin packing. Sci. China Inf. Sci. 65, 112105 (2021).
19. Galrão Ramos, A., Oliveira, J.F., Gonçalves, J.F., Lopes, M.P.: A container loading algorithm with static mechanical equilibrium stability constraints. Transp. Res. Part B Methodol. 91, 565–581 (2016).
20. George, J.A., Robinson, D.F.: A heuristic for packing boxes into a container. Comput. Oper. Res. 7, 147–156 (1980).
21. Zhao, A., Li, T., Lin, L.: A Dynamic Multi-modal deep Reinforcement Learning framework for 3D Bin Packing Problem. Knowl.-Based Syst. 299, 111990 (2024).
22. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization, http://arxiv.org/abs/1412.6980, (2017).
23. Li, D., Ren, C., Gu, Z., Wang, Y., Lau, F.: Solving Packing Problems by Conditional Query Learning. (2019).
24. Wu, Y., Li, W., Goh, M., de Souza, R.: Three-dimensional bin packing problem with variable bin height. Eur. J. Oper. Res. 202, 347–355 (2010).