Prioritized Experience Replay Based on dynamics priority

Hu Li (6213114015@stu.jiangnan.edu.cn)
Jiangnan University
Xuezhong Qian
Jiangnan University
Wei Song
Jiangnan University

Article

Keywords:

Posted Date: December 15th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3440928/v1

Additional Declarations: No competing interests reported.

Prioritized Experience Replay Based on dynamics priority

Hu Li[1,*], **Xuezhong Qian**[2], **and Wei Song**[1,2,+]

[1]jiangnan university, School of Artificial Intelligence and Computer Science, Wuxi, 214122, China
[2]jiangnan university, School of Artificial Intelligence and Computer Science, Wuxi, 214122, China
[*]6213114015@stu.jiangnan.edu.cn
[+]these authors contributed equally to this work

ABSTRACT

Experience replay has been instrumental in achieving significant advancements in reinforcement learning by increasing the utilization of data. To further improve the sampling efficiency, prioritized experience replay (PER) was proposed. This algorithm prioritizes experiences based on the temporal difference error (TD error), enabling the agent to learn from more valuable experiences stored in the experience pool. While various prioritized algorithms have been proposed, they ignored the dynamic changes of experience value during the training process, merely combining different priority criteria in a fixed or linear manner. In this paper, we present a novel prioritized experience replay algorithm called PERDP, which employs a dynamic priority adjustment framework. PERDP adaptively adjusts the weights of each criterion based on average priority level of the experience pool and evaluates experiences' value according to current network. We apply this algorithm to the SAC model and conduct experiments in the OpenAI Gym experimental environment. The experiment results demonstrate that the PERDP exhibits superior convergence speed when compared to the PER.

Introduction

Reinforcement learning is an approach to sequential decision making aimed at optimizing an agent's learning strategy during its interactions with the environment to achieve maximum expected cumulative reward[1]. In recent years, reinforcement learning has gained significant popularity across various domains. Different scenarios require the careful design of state spaces, action spaces, and rewards to effectively simulate the sequential decision making process inherent in reinforcement learning. Despite the remarkable successes achieved by reinforcement learning, challenges such as low data utilization efficiency and limited generalizability persist[2].

There is a notable issue with the continuous experimental samples get from the environment, as they tend to exhibit strong correlations, which contradicts the requirement of deep neural networks for independent and identically distributed data. To address this problem, Lin proposed the concept of experience replay, which has proven to be a reasonable solution[3][4].The experience replay mechanism stores experiences in an experience pool and reuses them. This approach effectively mitigates the issues of correlated distribution among samples, while also avoiding sample wastage. However, the sampling strategy of this method is uniform sampling, meaning that it assigns the same probability to each sample without considering their relative importance. In practice, agents often benefit more from learning important experiences.

Since then, many prioritized sampling strategies have been proposed. One of the most well-known is the prioritized experience replay (PER) introduced by Schaul[5], which improved by experience replay. Schaul used the time difference error (TD error) as a priority criterion to determine the importance of experiences. Subsequent experimental results combining various models have shown that PER significantly improves the sampling efficiency compared to traditional experience replay[6]. Subsequently, more modifications and improvements have been made to the PER algorithm. Ramicic, for instance, employed state entropy as a prioritization criterion, reasoning that experiences containing more unknown information have greater learning potential[7]. Li proposed three value measures for experiences and prioritied them, with experimental results validating that the TD error serves as an upper bound for these three metrics[8]. Shivakanth proposed a sampling strategy based on the learnability of samples, effectively avoiding repetition of training experience and reducing noise to improve learning efficiency[9].

The above-mentioned algorithms have all used only single priority criterion, which has some drawbacks. Firstly, it is highly sensitive to environmental changes, which performing unstable in different environments[10]. Secondly, evaluating all experiences accurately becomes challenging due to the large capacity of the experience pool[11]. In response to these challenges, Xi Cao proposed a strategy called High Value Prioritized Experience Replay (HVPER), which combines the state value function and TD error as priority criteria. HVPER mitigates the negative impact of experience with high TD errors near the edge of state space, thereby enhancing learning efficiency and accelerating algorithm convergence[12]. Jiashan Gao considers the immediate reward of experiences and incorporates it as a criterion by linearly combining it with the TD error[13]. To prevent valuable old experiences in the experience pool from being under-sampled for an extended period, Ximing Liu introduced a dynamic experience replay strategy known as PERMAB[14]. This method aims to dynamically evaluate the contribution of each priority during the training process to prevent the model from achieving a suboptimal performance. However, PERMAB still calculates priority scores in a linear manner and only applies non-linear weighting during the calculation of priority weights.

In this paper, we present a novel approach that improves the efficiency for experience replay. Our method incorporates dynamic adjustment of priority criterion weights during the training process, taking into account recent feedback information. The agent estimates average priority level of the experience pool based on the interaction between sampled experiences and current network. During next training iteration, the sampled experience priority weight are re-evaluated according to the average priority level. We dynamically consider the significance of different criteria and update the priorities based on the adjusted weights.

1 Method

1.1 Preliminary knowledge

1.1.1 Knowledge

In online reinforcement learning, updating the network after each interaction with the environment can result in catastrophic forgetting of past experiences. Additionally, the correlation between successive exploration data also brings a challenge to network update. To address these issues, Lin introduced an experience replay mechanism, which storing observed experiences in an experience pool and sampling batches from it during training. This approach has been widely successful when combined with various reinforcement learning algorithms[15].

Lin's experience replay method employs a uniform sampling strategy that treats all experiences equally, which can be considered unreasonable. To address this limitation, Schaul introduced the Prioritized Experience Replay (PER) algorithm, which assigns individual priorities to each experience stored in the experience buffer. In PER, Schaul utilizes TD error to quantify the amount of information contained in an experience and promotes prioritized training for experiences with higher unknown information. The TD error is defined by the following equation:

$$\delta = R(s,a,s') + \gamma \cdot v(s') - v(s) \tag{1}$$

where R(s,a, s') represents the immediate rewards, s and s' denote the current state and next states, v(s) is the estimated state value function for state s, and γ is the discount factor.

To ensure that every experience is sampled, PER implements a stochastic prioritization approach rather than a greedy strategy. This is important as a greedy strategy might neglect experiences with low TD error, causing them to be undersampled or even forgotten. The sampling probability P_i of an experience is calculated using the following equation:

$$P_i = \frac{{p_i}^{\alpha}}{\sum {p_k}^{\alpha}} \tag{2}$$

where, $p_i = |\delta| + \varepsilon$ represents the priority of experience i. α is used to adjust the degree of priority and ε is used to prevent the probability from being zero.

This priority-based sampling method may changes the probability distribution of experiences, introducing a bias in estimating the action value function Q(s, a). To address this bias, PER uses importance sampling and the importance-sampling weight ω_i for experience i is demonstrated by the following equation:

$$\omega_i = \left(\frac{1}{N} \cdot \frac{1}{P_i}\right)^{\beta} \tag{3}$$

where, β is another hyperparameter that controls the degree of correcting bias.

1.1.2 Problem statement

The priority of samples in the experience pool is dynamically changing. With the implementation of stochastic prioritization strategy, all batches will eventually converge, as demonstrated in the figure$1-2$. Nevertheless, different criteria will converge at different stages of training process. It can be observed that the TD-error tends to stabilize after 30 episodes, while the reward converges much earlier.

Efficient sampling experiences from the experience replay buffer is a key challenge in experience replay. To address this issue, we propose an experience replay algorithm evaluating comprehensively the value of experience. By doing so, the agent can sample more valuable experiences from the experience replay buffer B, thereby accelerating the learning speed.

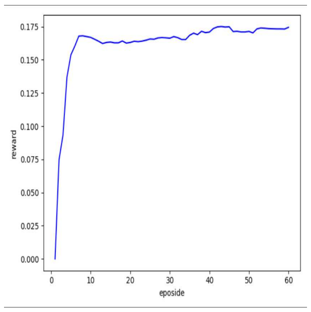

Figure 1. The reward over samples batch with eposidos

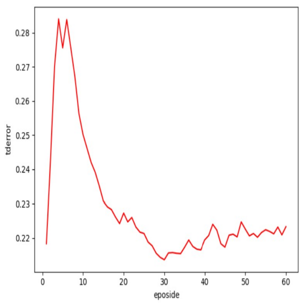

Figure 2. The td-error over samples batch with eposidos

1.2 Improved method

In this section, we introduce the PERDP algorithm, which assesses the value of experiences by considering two prioritization criteria: TD-error and reward. This algorithm adjusts the weights of each criterion with respect to the current network.

PERDP uses TD error as a priority because it reflects the uncertainty of an experience. Experiences with higher TD errors indicate a larger discrepancy between the predicted Q-values by the current network, indicating their higher value. The calculation of TD error priority in PERDP is as follows:

$$p_t = |R_t + \gamma \cdot \max_{a'} Q(S_t, a') - Q(S_{t-1}, A_{t-1})| \tag{4}$$

where, R_t represents the immediate reward at time step t, γ is the discount factor, Q(S_t, a') denotes the Q-value for state-action pair (S_t, a'), and Q(S_{t-1}, A_{t-1}) represents the Q-value predicted for the previous state-action pair (S_{t-1}, A_{t-1}).

Despite the advantages of PER algorithm over uniform sampling, it still suffers from some drawbacks. One limitation is that the priorities of experiences in the replay buffer often become outdated because updating the priority of every experience is impractical[16][17]. Additionally, continually sampling high priority experiences can lead to overfitting[18]. Relying solely on a single perspective when evaluating experiences may restrict the model to local optimum[19]. Furthermore, high TD-error experiences are often appear near the edge of state space, as the agent rarely explore in these areas. However, this does not necessarily mean that these experiences can provide significant useful information.

To address this issue, we use another widely used priority criterion called immediate reward p_r, mitigating bias through multiple criteria. Immediate Rewards are the most direct reflection of the interaction between agent and environment, especially in sparse reward environments. By considering p_r, the agent can account for the historical value of an experience. The calculation of p_r is as follows:

$$p_r = R(s, a, s') + \varepsilon \tag{5}$$

where, R(s, a, s') represents the immediate reward, and ε is a small positive constant that ensures all experiences have non-zero probabilities of being selected.

During different stages of training, the influence of p_R and p_t on the current network constantly changes. However, specifying fixed or linear priority weights artificially would ignore interaction between agent and environment. To accurately evaluate the value of experiences at different stages, we propose a mechanism for updating dynamically priority weights. This method adjusts the weights based on the importance of each criterion to the experience pool.

We sample a batch of size M from the experience pool t times and the total number of experiences available is N. To determine the priority of each sampled experience, we first calculate p_r and p_t based on the current network. Then, we get the actual score μ_j for each recent priority, which represents the average priority level of the experience pool. The calculation of μ_j is as follows:

$$\mu_j = \frac{\sum_{i=1}^{T} \delta_{ij}}{N} \tag{6}$$

where, δ_{ij} is the priority of the i-th experience under the j-th criterion. For the newly sampled batch, the importance S_j for each priority criterion is calculated based on the average priority level of the experience pool μ_j with the following formula:

$$S_j = P\ (\ \delta_{ij}\ >=\ \mu_j) \tag{7}$$

where, S_j is the estimation of the advantage probability of each priority criterion relative to average priority level. If the probability of a criterion in the batch is higher than the feedback informationμ_j, it indicates that the criterion is currently more important compared to others.

Based on S_j, we further calculate the influence factor F_j for each priority criterion as follows:

$$F_j = e^{dS_j} - 1 \tag{8}$$

We calculate the weight W_j for each criterion according to F_j. The calculation formula is as follows:

$$W_j = \frac{F_j}{\sum_{k=1}^{N} F_k} \tag{9}$$

By using this formula, we can get the weight W_j for each criterion, which reflects the relative importance of each criterion in the learning process. This allows the agent to prioritize and focus on more meaningful experience, enabling effective learning.

After the new priority weights have been calculated, the experience is re-prioritized and saved to the experience pool. The new priority calculation formula is as follows:

$$P = \sum_j p_j \cdot W_j \tag{10}$$

In summary, we propose an experience replay algorithm that combines TD-error and immediate reward with dynamically updated weights. When a batch is sampled from the experience pool, the agent updates the network parameters and evaluates the relative importance of each priority criterion in the context of the current network and the average level of the experience pool. After calculating the priority weights, the experience priority is updated. The detailed algorithm is shown in Algorithm 1.

Algorithm 1 *PERDP*

```
Require: discount factor $\gamma$, learning rate $\alpha$, total steps S, memory size M, batch size B
Ensure: Initialize the environment, network parameters$theta$, target network parameters $\theta_t$
 1: while $timesteps >= 0$ do
 2:     observe the state $s_t$, select the action $a_t$
 3:     execute the action $a_t$ and get the reward $r_t$ , the next state $s_{t+1}$
 4:     Store transition $(s_t, a_t, s_{t+1}, r_t)$ into experience buffer
 5:     if update network then
 6:         Sample a batch of N transition from experience buffer
 7:         Update priority weight $W_j$ according to Eq.7-9
 8:         Update network parameters $\theta$ , $\theta_t$
 9:         calculate experience priority P according to Eq.10 and Store it
10:         Compute $\mu_j$ according to Eq.4-6
11:     end if
12:     if $timestep = S$ then
13:         break
14:     end if
15: end while
```

Condition	PERDP	PER	RT
MsPacman	55.75k	69.96k	73.36k
Alien	42.24k	59.63k	70.55k
UpNDone	16.87k	27.27k	46.18k

Table 1. Comparison of time steps required for model convergence.

2 Experience

We apply the proposed algorithm, PERDP , to the SAC model and conduct experiments in video games environment. The objective is to validate that PERDP can converge rapidly and achieve optimal performance with fewer training steps. For comparison, we select two improved algorithms for DQN as baselines: PER[5] and RT (Reward and td error parameter) experience replay[13]. In total, three algorithms are appear in the experiment, as follows:

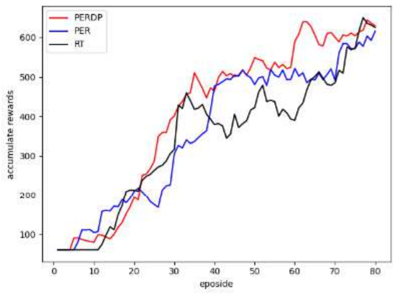

Figure 3. evaluation results of the MsPacman environment

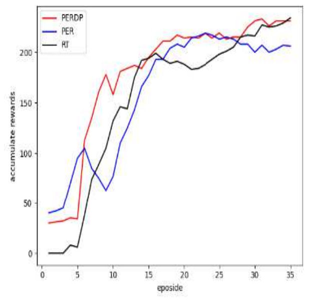

Figure 4. evaluation results of the Alien environment

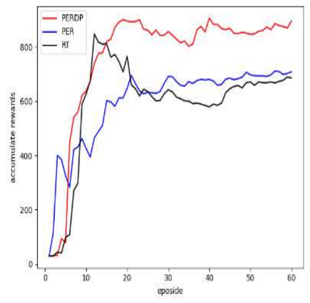

Figure 5. evaluation results of the UpnDown environment

(1) PER-prop: This algorithm prioritizes experiences based on TD-error, which is a modified version of the DQN algorithm with uniform sampling. The TD-error serves as an indicator of the importance of an experience.

(2) RT: This method is a prioritized experience replay algorithm that combines TD-error and reward. The weight between these priorities is fixed.

(3) PERDP: This approach is an improved algorithm of RT, which adjusts dynamically the weights .

2.1 Experience detail

To ensure the fairness of the experiments, we conducted all algorithms in the same environment with identical hyperparameters. Our algorithm is implemented on the basis of SAC-Discrete using PyTorch. The following hyperparameters were used:learning rate is 0.0003, gamma is 0.99, hidden layer of policy network is a fully connected neural network with 512 neurons, number of input neurons is state space, number of output neurons is action space.In contrast, the batch size was set to 32, the experience pool capacity to 50,000, and the target network update interval to 1,000 time steps. Additionally, each algorithm use a soft update strategy. SAC-Discrete using PyTorch code is vailable from https://github.com/ku2482/sac-discrete.pytorch.

2.2 Atari experience

2.2.1 Experimental results

We evaluated the PERDP algorithm in three Atari environments. The cumulative reward evaluation results of the three algorithms are shown in Figure3 – 5. As can be seen, PERDP algorithm outperforms the other two methods overall, achieving higher cumulative rewards. Although our algorithm initially did not perform as well and had results similar to the other two algorithms, it gradually showed its advantages as the training going on. This is because in the early stages, the agent receives sparse rewards with a lot of noise, causing it difficult for PERDP to measure the overall value of the experience pool. As the experience pool accumulates more valuable experiences,the learning speed of the agent increases rapidly, allowing it to quickly learn the optimal policy. Both PER and RT show the same trend, but PERDP has significant advantages in the later stage of the training process. Table 1 indicates that PERDP achieves good performance with fewer steps.

The PER algorithm obtains rewards quickly in the early stages, but the learning speed slows down in the mid-term stage. This is because in the early stages, the ability of the TD-error to explore unknown information allows the model to quickly acquire advantageous experiences. as the policy gradually approaches the optimal network, PER does not select more importance experiences. RT algorithm is similar to PERDP and can learn quickly. However, it performs worse in the later stage because the fixed-weight approach fail to sample experiences based on environmental change. Human guidance cannot be maintained throughout the entire training process, so some "outdated" experiences may not have a positive impact and cause the training to be extremely unstable.

2.2.2 Result analysis

In order to better understand the PERDP algorithm, we analyzed the average reward and state entropy. Figure 6 illustrates the average reward obtained by the agent. It can be seen that the PER achieves higher average rewards in the early stages due to its focus on exploring the positional information of the environment. However, as training going on, TD error becomes less important for policy networks that have predictive ability, instead performing similar to uniform sampling. On the other hand, PERDP evaluates the priority of experiences from multiple perspectives in the later stages and replay the most valuable experiences from the experience pool. This enables the model to maintain an advantage over the other two methods.In different environments, each standard priority tends to stabilize, similar to uniform sampling. However, PERDP can assess the contribution value of priority standards, enabling the agent to learn more.

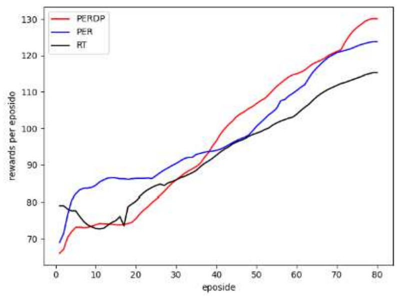

Figure 6. this is a comparison of the average rewards among three methods.

The curve depicting the change of state entropy over the training steps is shown in Figure 7. It can be observed that the state entropy of PERDP is higher than the other two methods in the initial stages, while PER and RT exhibit similar trends. This indicates that our approach ensures that the agent encourages exploration. Subsequently, the state entropies of the methods gradually decrease and approach each other as the networks begin to have some predictive ability.

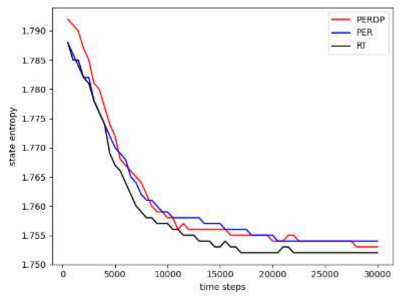

Figure 7. this is a comparison of the entropy among three methods.

3 Discussion and Conclusion

In this study, we introduce a novel method called Prioritized experience replay with dynamics priority (PERDP) to enhance sampling efficiency. PERDP combines TD-error and immediate reward to select experiences from the experience pool. By analyzing the importance of both priority criterion, weights are assigned based on the current experience pool when calculating the total priority. This comprehensive approach allows for a thorough assessment of experience value from multiple perspectives, considering the impact of different priority criteria on various types of experiences during training. Consequently, there is no need for artificial judgment of priority criterion.The experimental results obtained from a discrete action control task demonstrate the superior performance of the PERDP method. In the future, we plan to explore other efficient priority criteria and further categorize experiences in more detail.

References

1. Li Y. Deep reinforcement learning: An overview[J]. arXiv preprint arXiv:1701.07274, 2017.

2. Laskin M, Lee K, Stooke A, et al. Reinforcement learning with augmented data[J]. Advances in neural information processing systems, 2020, 33: 19884-19895.

3. Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning,planning and teaching. Machine Learning,8(3-4),293?321. http://dx.doi.org/10.1007/BF00992699

4. Fedus W, Ramachandran P, Agarwal R, Bengio Y, Larochelle H, Rowland M, Dabney W. (2020). Revisiting Fundamentals of Experience Replay. Proceedings of the 37th International Conference on Machine Learning, Proceedings of Machine Learning Research 119:3061-3071

5. Schaul T, Quan J, Antonoglou I, Silver D. (2016). Prioritized experience replay.In Proceedings of the international conference on learning representations (ICLR).

6. Hou Y , Liu L , Wei Q , et al. A novel DDPG method with prioritized experience replay[C]// IEEE International Conference on Systems. IEEE, 2017.

7. M. Ramicic and A. Bonarini, "Entropy-based prioritized sampling in Deep Q-learning," 2017 2nd International Conference on Image, Vision and Computing (ICIVC), Chengdu, China, 2017, pp. 1068-1072, doi: 10.1109/ICIVC.2017.7984718.

8. Ang A. Li, Zongqing Lu, Chenglin Miao(2021). Revisiting Prioritized Experience Replay: A Value Perspective. ArXiv.arXiv:2102.03261.

9. Shivakanth Sujit, Somjit Nath, Pedro H. M. Braga, Samira Ebrahimi Kahou (2022). Prioritizing Samples in Reinforcement Learning with Reducible Loss. ArXiv.arXiv:2208.10483.

10. Novati, G; Koumoutsakos, P. (2019). Remember and Forget for Experience Replay. Proceedings of the 36th International Conference on Machine Learning, Proceedings of Machine Learning Research 97:4851-4860

11. Daochen Zha , Kwei-Herng Lai , Kaixiong Zhou , Xia Hu(2019). Experience Replay Optimization. ArXiv.arXiv:1906.08387

12. X. Cao, H. Wan, Y. Lin , S. Han, "High-Value Prioritized Experience Replay for Off-Policy Reinforcement Learning," 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 2019, pp. 1510-1514, doi: 10.1109/ICTAI.2019.00215.

13. J. Gao, X. Li, W. Liu , J. Zhao, "Prioritized Experience Replay Method Based on Experience Reward," 2021 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE), Chongqing, China, 2021, pp. 214-219, doi: 10.1109/MLISE54096.2021.00045. [08]

14. Ximing Liu, Tianqing Zhu, Cuiqing Jiang, Dayong Ye, Fuqing Zhao, Prioritized Experience Replay based on Multi-armed Bandit, Expert Systems with Applications, Volume 189, 2022, 116023, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.116023.

15. Ziyu Wang, Victor Bapst, Nicolas Hees, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, Nando de Freitas(2017). Sample Efficient Actor-Critic With Experience Replay. ArXiv.arXiv:1611.01224.

16. Pan. Y, Mei. J, Farahmand. A, White. M, Yao. H, Rohani. M, Luo. J. (2022). Understanding and mitigating the limitations of prioritized experience replay. Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence, in Proceedings of Machine Learning Research 180:1561-1571 Available from https://proceedings.mlr.press/v180/pan22a.html.

17. Min Li, Tianyi Huang, William Zhu, Clustering experience replay for the effective exploitation in reinforcement learning, Pattern Recognition, Volume 131, 2022, 108875, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2022.108875.

18. P. Buzzega, M. Boschini, A. Porrello and S. Calderara, "Rethinking Experience Replay: a Bag of Tricks for Continual Learning," 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 2021, pp. 2180-2187, doi: 10.1109/ICPR48806.2021.9412614.

19. Shangtong Zhang, Richard S. Sutton(2018). A Deeper Look at Experience Replay. arXiv preprint arXiv: 1712.01275.

Author contributions statement

Hu Li: Conceptualization, Methodology, Writing - original draft . **XueZhong Qian**: review , editing. **Wei Song**: Supervision.

Availability of data and materials

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Funding

This work was supported by the National Natural Science Foundation of China (62076110), the Natural Science Foundation of Jiangsu Province (BK20181341).

Competing Interests

The authors declare that they have no confict of interest.

Ethics approval

This article does not involve any studies conducted by the authors with human participants or animals.