frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 09 November 2023
DOI 10.3389/fphy.2023.1310467

OPEN ACCESS

EDITED BY
Dun Han,
Jiangsu University, China

REVIEWED BY
Peng Hu,
Anhui Normal University, China
Ruxin Zhao,
Yangzhou University, China

*CORRESPONDENCE
Hengfu Yin,
✉ hfyin@caf.ac.cn
Yupeng Wang,
✉ yupengwang@njust.edu.cn

RECEIVED 09 October 2023
ACCEPTED 24 October 2023
PUBLISHED 09 November 2023

CITATION
Sang J, Ahmad Khan Z, Yin H and Wang Y (2023), Reward shaping using directed graph convolution neural networks for reinforcement learning and games.
Front. Phys. 11:1310467.
doi: 10.3389/fphy.2023.1310467

Reward shaping using directed graph convolution neural networks for reinforcement learning and games

Jianghui Sang[1,2], Zaki Ahmad Khan[3], Hengfu Yin[1]* and Yupeng Wang[2]*

[1]Research Institute of Subtropical Forestry, Chinese Academy of Forestry, Hangzhou, China, [2]School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China, [3]Department of Computer Science, University of Worcester, Worcester, United Kingdom

Game theory can employ reinforcement learning algorithms to identify the optimal policy or equilibrium solution. Potential-based reward shaping (PBRS) methods are prevalently used for accelerating reinforcement learning, ensuring the optimal policy remains consistent. Existing PBRS research performs message passing based on graph convolution neural networks (GCNs) to propagate information from rewarding states. However, in an irreversible time-series reinforcement learning problem, undirected graphs will not only mislead message-passing schemes but also lose a distinctive direction structure. In this paper, a novel approach called directed graph convolution neural networks for reward shaping φ_{DCN} has been proposed to tackle this problem. The key innovation of φ_{DCN} is the extension of spectral-based undirected graph convolution to directed graphs. Messages can be efficiently propagated by leveraging a directed graph Laplacian as a substitute for the state transition matrix. As a consequence, potential-based reward shaping can then be implemented by the propagated messages. The incorporation of temporal dependencies between states makes φ_{DCN} more suitable for real-world scenarios than existing potential-based reward shaping methods based on undirected graph convolutional networks. Preliminary experiments demonstrate that the proposed φ_{DCN} exhibits a substantial improvement compared to other competing algorithms on both Atari and MuJoCo benchmarks.

KEYWORDS

Markov decision process, reinforcement learning, directed graph convolutional network, reward shaping, game

1 Introduction

Over the past few decades, game theory has utilized the concepts and methods of reinforcement learning (RL) to solve decision-making problems [1]; [2]; [3]. An RL problem can be seen as a game between individual decision-makers and the environment [4]; [5]. RL can be expressed as a Markov decision process (MDP) [6]; [7]. Through interaction between agents and the environment, RL is able to receive rewards and take actions to maximize those rewards. During the training of RL, agents often encounter situations where they cannot obtain rewards most of the time [8]; [9]. Providing rewards in a sparse environment makes learning difficult for agents [10]. It is important to note that RL has always been hampered by sparse rewards [11].

Reward shaping is a widely used technique to address the challenge of reward sparsity. The purpose of reward shaping is to guide agents in learning through providing artificially designed additional rewards. Nevertheless, artificially designed reward functions may result in agents learning non-optimal policies in certain situations. Therefore, potential-based reward shaping (PBRS) is proposed in literature [12]. In this manner, the optimal policy is maintained when the additional reward value can be expressed in the differential form of a potential function. Thus, PBRS effectively avoids the reward hacking problem while addressing the sparse reward issue. On the other hand, RL problems can be considered probabilistic inference problems in hidden Markov models, where forward–backward messages can be used for inference. According to existing research, the potential function is defined in the probabilistic inference view of RL. The probability of an optimal trajectory is usually defined as a potential function under a given state. Due to the complexity of computation, it is difficult to obtain the messages. As a consequence, reward shaping using graph convolution networks is developed since projections of functions on the eigenspace of the graph Laplacian produce smooth approximation with respect to the underlying state-space topology of the MDP.

Previous research relies on a traditional spectral-based GCN to leverage reward shaping [13], but often overlooks crucial temporal dependencies between states. Sami et al. [14] employed a recurrent neural network to record reward-shaping values at different times, but the issue of temporal dependencies is not addressed essentially. Their use of the undirected graph and symmetric Laplacian matrix resulted in messages being interfered with and discarded in different directions, which may lead to serious logical errors. From a macro perspective, there are indeed some tracks that are sequential and irreversible in the real world.

To tackle the aforementioned problem, we propose an approach termed directed graph convolution neural networks for reward shaping φ_{DCN}. Our approach extends spectral-based undirected graph convolution to a directed graph which is built on a state probability model of trajectory. The state transition matrix is approximated by a directed graph Laplacian in the process of reward shaping. Messages about states that are propagated on this directed graph Laplacian serve to learn potential functions. The incorporation of temporal dependencies between states makes φ_{DCN} more suitable for real-world scenarios than existing potential-based reward-shaping methods based on undirected graph convolutional networks. We have demonstrated that φ_{DCN} outperforms competitive baselines on both Atari [15] and [16] benchmarks.

The main contributions are summarized as follows.

- We implement reward shaping through the message-passing mechanism of directed graph neural networks for the first time, which is more in line with the logic of the real world.
- The stationary distribution of the classical directed graph the Markov chain builds on is not necessarily unique since the graph might not be necessarily irreducible and aperiodic. To counteract this, we added a PageRank-based teleportation back to each node.
- Experiments demonstrate that the performance of the proposed φ_{DCN} exceeds that of the baseline algorithm on the Atari and MuJoCo benchmark.

2 Related work

2.1 Reward shaping

Reward shaping is used to accelerate learning when the environment only provides sporadic, incomplete, or delayed rewards. Ng et al. [12] proposed an extended version called potential-based reward shaping. Its most acclaimed characteristic is its ability to ensure that the optimal policy remains unchanged, as supported [12]. On this basis, there are two development paths of potential-based reward shaping: a) potential-based advice [17]; [18] and b) dynamic potential-based reward shaping [19]. Potential-based advice adds state–action pairs into potential functions rather than individual states. It is possible to vary the potential energy function over time using the latter approach. [20] is a book about reward shaping, which provides a detailed summary of the methods, theories, and applications of reward shaping.

In addition to the potential-based reward-shaping methods, other outstanding research studies on reward shaping also embrace belief reward shaping, ethics shaping, and reward shaping via meta-learning. These works are different from ours as we utilize convolutional networks to leverage reward shaping. In this sense, we focus more on the message-passing mechanism in the network to learn the potential function.

2.2 Digraph convolution

Digraph convolution is a method for performing convolution operations on a directed graph, which aims to comprehensively analyze the topological structure of the graph and the feature information of nodes or edges. Compared to undirected graph convolutional networks, digraph convolutional networks have the advantage of better reflecting the directional relationships between nodes. [21] extended the graph convolutional kernel originally designed for an undirected graph to a directed graph by introducing a trainable binary gating mechanism. This enables the model to regulate information dissemination based on the directionality of edges. Moreover, Li et al. proposed a new neural network architecture that can directly process graph structured data [22], including graph convolution operations for directed graphs. [23] proposed a new graph convolution method called MixHop, which can simultaneously consider the information of all neighboring nodes and handle directed graphs.

Some GCNs are designed to adapt to directed graphs (digraphs) by looking for structural patterns and reformulating the graph [24]; [25]. In addition to their limitations, these methods rely on pre-defined structures and are not capable of handling complex structures. Similar to our approach, another approach [26] redefines the propagation scheme only for strongly connected digraphs. In contrast, our approach is universally applicable to digraphs, which is the most important difference.

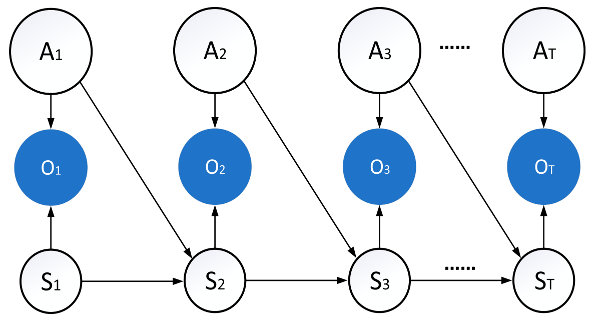

FIGURE 1
Graphical model with the variable O_t.

3 Background

3.1 Basic notions

As a mathematical expression, an MDP is represented by the tuple $\langle S, A, \gamma, r, P \rangle$, where S is the state space, A is the action space, r is the reward function, P is the transition probability matrix with $P(s' \mid s, a)$ giving the probability of transitioning to state s' when action a is taken at state s, and $\gamma \in (0, 1)$ is the discount factor. The state–action trajectory of a policy can be modeled by $\tau = (s_0, a_0, s_1, a_1 \ldots)$.

The policy π value function is defined as follows:

$$V_r^{\pi}(s) = \mathrm{E}_{\tau\sim\pi}\left[\sum\nolimits_{t=0}^{\infty}\gamma^t r(s_t, a_t) \mid s_0 = s\right]. \tag{1}$$

The policy π action-value function is denoted as follows:

$$\mathrm{Q}_r^{\pi}(s, a) = \mathrm{E}_{\tau\sim\pi}\left[\sum\nolimits_{t=0}^{\infty}\gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a\right]. \tag{2}$$

The policy π expected discounted return is defined as follows:

$$J(\pi) = \mathrm{E}_{\tau\sim\pi}\left[\sum\nolimits_{t=0}^{\infty}\gamma^t r(s_t, a_t)\right]. \tag{3}$$

Algorithms that learn from RL can determine the optimal policy π^*:

$$\pi^* = \mathrm{argmax}_{\pi} J(\pi). \tag{4}$$

Hence, given the initial state s_0 and its distribution $d\ (s_0)$, the gradient of the $J(\pi)$ over a parameterized policy π_θ can be expressed as

$$\begin{aligned} \frac{\partial J(\theta)}{\partial \theta} &= \sum\nolimits_s d(s;\theta) \sum\nolimits_a \frac{\partial \pi_\theta(a \mid s)}{\partial \theta} \mathrm{Q}_{\pi_\theta}(s, a), \\ d(s;\theta) &= \sum\nolimits_{s_0} d(s_0) \sum\nolimits_{t=0}^{\infty} \gamma^t P^{\pi_\theta}(S_t = s \mid S_0 = s_0). \end{aligned} \tag{5}$$

3.2 Control as inference

In existing studies, the RL problems have been translated directly into probabilistic inference problems [27]; [28]. So we use probability graph models to approximate RL. We then implement probabilistic inference through a message-passing mechanism. When the states are represented by nodes and the edges represent the transition probability between two different states, MDPs were considered probability graph models in previous RL research studies. As shown in Figure 1, the RL structure approximates hidden Markov models (HMMs). Taking this into account, we introduce a binary variable $O = 0$ or 1 based on the action A_t and the state S_t. When $O_t = 1$, the state–action pair is optimal in time t.

In the probabilistic inference view of RL, the value function $V_\pi(S)$ can be approximately inferred through a message-passing mechanism. The forward–backward algorithm is an effective approach for performing inference in an HMM. A backward message is defined as $\beta(S_t, A_t) = P(O_{t:\,T} | S_t, A_t)$, and a forward message is defined as $\alpha(S_t, A_t) = P(O_{0:\,t-1} | S_t, A_t) P(S_t, A_t)$, where $O_{t:T}$ is the observation variable from time t to the end. Correspondingly, $O_{0:t-1}$ is the observation variable from the beginning to time $t-1$. In the RL graph, given the current state S_t, the backward message reflects the probability that the current trajectory will lead to an optimal one in the future. This forward message indicates the probability of a past optimal trajectory for the current state S_t. By projecting maximum-entropy RL into probability space, the mapping function $f(\cdot)$ can be determined. $f(\cdot)$ maps rewards to a probability space by defining the distribution of this optimality variable as $P(O = 1 | S_t, A_t) = f(r(S_t, A_t))$.

As a result of recursion, the forward $\alpha(S_t, A_t)$ and backward $\beta(S_t, A_t)$ messages can be expressed as follows:

$$\begin{aligned} \alpha(S_t, A_t) &= \sum_{S_{t-1}} \sum_{A_{t-1}} P(S_t \mid S_{t-1}, A_{t-1}) P(A_t) P(O_{t-1} \mid S_{t-1}, A_{t-1}) \alpha(S_{t-1}, A_{t-1}), \\ \beta(S_t, A_t) &= \sum_{S_{t+1}} \sum_{A_{t+1}} P(S_{t+1} \mid S_t, A_t) P(A_{t+1}) P(O_t \mid S_t, A_t) \beta(S_{t+1}, A_{t+1}). \end{aligned} \tag{6}$$

It should be noted that only the current state and reward are visible, not the action space. So the potential function of PBRS is designed in the state space. Once the actions are marginalized, we redefine the forward message $\alpha(S_t)$ and backward message $\beta(S_t)$ for learning potential functions.

According to [12], the optimal policy will remain unchanged after φ_{DCN} implements potential-based reward shaping. By replacing the original reward function $r\ (S_t, A_t)$ with a new reward function $R\ (S_t, A_t, S_{t+1})$, potential-based reward shaping can guarantee the optimal policy unchanged in RL:

$$R(S_t, A_t, S_{t+1}) = r(S_t, A_t) + F(S_t, S_{t+1}). \tag{7}$$

Here, $F(S_t, S_{t+1})$ is the shaping function calculated as follows:

$$F(S_t, S_{t+1}) = \gamma \Phi_{\alpha\beta}(S_{t+1}) - \Phi_{\alpha\beta}(S_t). \tag{8}$$

Literature reports have shown that the propagated messages can be used as potential functions [13]; [29]. Given the marginalized messages $\alpha(S_t)$ and $\beta(S_t)$, the potential function $\Phi(\cdot)$ is defined as

$$\Phi_{\alpha\beta}(S_t) = \alpha(S_t)\beta(S_t). \tag{9}$$

The potential function represents the probability of an entire trajectory being optimal. A high-return pathway is indicated by their likelihood.

4 Directed graph convolution neural networks for reward shaping

A DCN utilizes directed graph convolution to propagate messages $\alpha(S_t)$ and $\beta(S_t)$. Here, we illustrate digraph Laplacian,

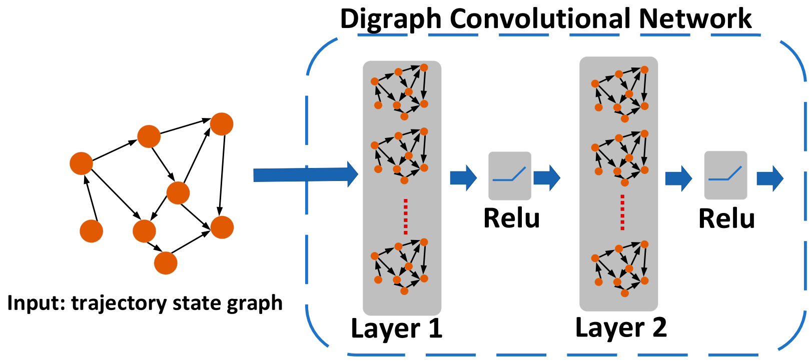

FIGURE 2
Network structure in φ_{DCN}. The trajectory in the MDP is defined as a graph, where each state is a node. The edge is the transition probability between two nodes. This probability directed graph is the input of φ_{DCN}.

TABLE 1 Configuration for experiments.

Hyperparameter	Value
Learning rate	2.5e-4
γ	0.99
Entropy coefficient	0.01
PPO steps	128
PPO clipping value	0.1
Mini batches	4
Processes	8
φ_{DCN}: μ	0.01
φ_{DCN}: λ	10
φ_{DCN}: δ	0.9

network structure, and loss function. In addition, we present a directed graph Laplacian. The structure of the directed graph convolutional neural network is shown in Figure 2.

4.1 Digraph Laplacian

The spectral-based graph convolution is first extended to the directed graph (digraph) by using the inherent relationship between the graph Laplacian and the stationary distribution of PageRank. Using the properties of Markov chains, we can solve the problem in digraphs using the internal relationship between graph Laplacian and PageRank. In a digraph $G = (V_d, E_d)$, a random walk on G is a Markov process with transition matrix $\mathrm{P}_{rw} = D_d^{-1} A_d$, in which $D_d(i,i) = \sum_j A_d(i,j)$ is the diagonal degree matrix and A_d is the adjacency matrix. The stationary distribution of PageRank may not be unique if the transition matrix is not necessarily irreducible and aperiodic, especially when a graph contains isolated nodes in the periphery or can be formed into a bipartite graph. Irreducibility means that there exists a path between any two nodes in the network, while aperiodicity means that the probability of returning to a node after a certain number of steps is not periodic. If a graph contains isolated nodes, then there is no path from those nodes to other nodes in the network, and the matrix is not irreducible. Similarly, if a graph can be formed into a bipartite graph, then there are no links between nodes in the same partition, which means that the matrix is not aperiodic.

Consequently, we slightly modify the random walk to PageRank which makes teleporting back to each node possible. In this way, the PageRank transition matrix P_{pr} is strictly irreducible and aperiodic, which is defined as $P_{pr} = (1-\mu)P_{pr} + \frac{\mu}{n}1^{n\times n}$. It should be noted that the variable μ is small enough. Thus, according to the Perron–Frobenius theory [30], P_{pr} has a unique left eigenvector ξ_{pr} which is strictly positive with eigenvalue 1.

The row-vector ξ_{pr} corresponds to the stationary distribution of P_{pr}, and we have $\xi_{pr}(i) = \sum_{i,i\rightarrow j}\xi_{pr}(i)P_{pr}(i,j)$. According to the equation, the ξ_{pr} of node i is the sum of all incoming probabilities from node j to node i. Therefore, the ξ_{pr} and an undirected graph degree matrix D_u have similar properties. The digraph Laplacian using PageRank ϕ_{pr} in symmetric normalized format is defined as

$$L_{pr} = I - \frac{1}{2}\left(\Pi_{pr}^{\frac{1}{2}} P_{pr} \Pi_{pr}^{-\frac{1}{2}} + \Pi_{pr}^{-\frac{1}{2}} P_{pr}^{T} \Pi_{pr}^{\frac{1}{2}}\right), \tag{10}$$

where we employ $\Pi_{pr} = \frac{1}{\|\xi_{pr}\|_1} Diag(\xi_{pr})$ to replace degree matrix D_u in an undirected graph. The definition is based on strongly connected digraphs, so it is not universally applicable. To deal with it, $\mu \rightarrow 0$ provides a generalized solution.

4.2 Loss function

One of the core characteristics of the GCN is that the message-passing mechanism is built on the graph Laplacian. Currently, PBRS is based on a traditional undirected GCN, where the undirected graph convolution is defined as $Z_u = \tilde{A}_u XW$. $\tilde{A}_u$ represents the normalized self-looped adjacency matrix of the undirected graph, and W represents the weight. The GCN and its variants require the undirected symmetric adjacency matrix A_u as the input. This not only aggregates features with incorrect weights but also discards

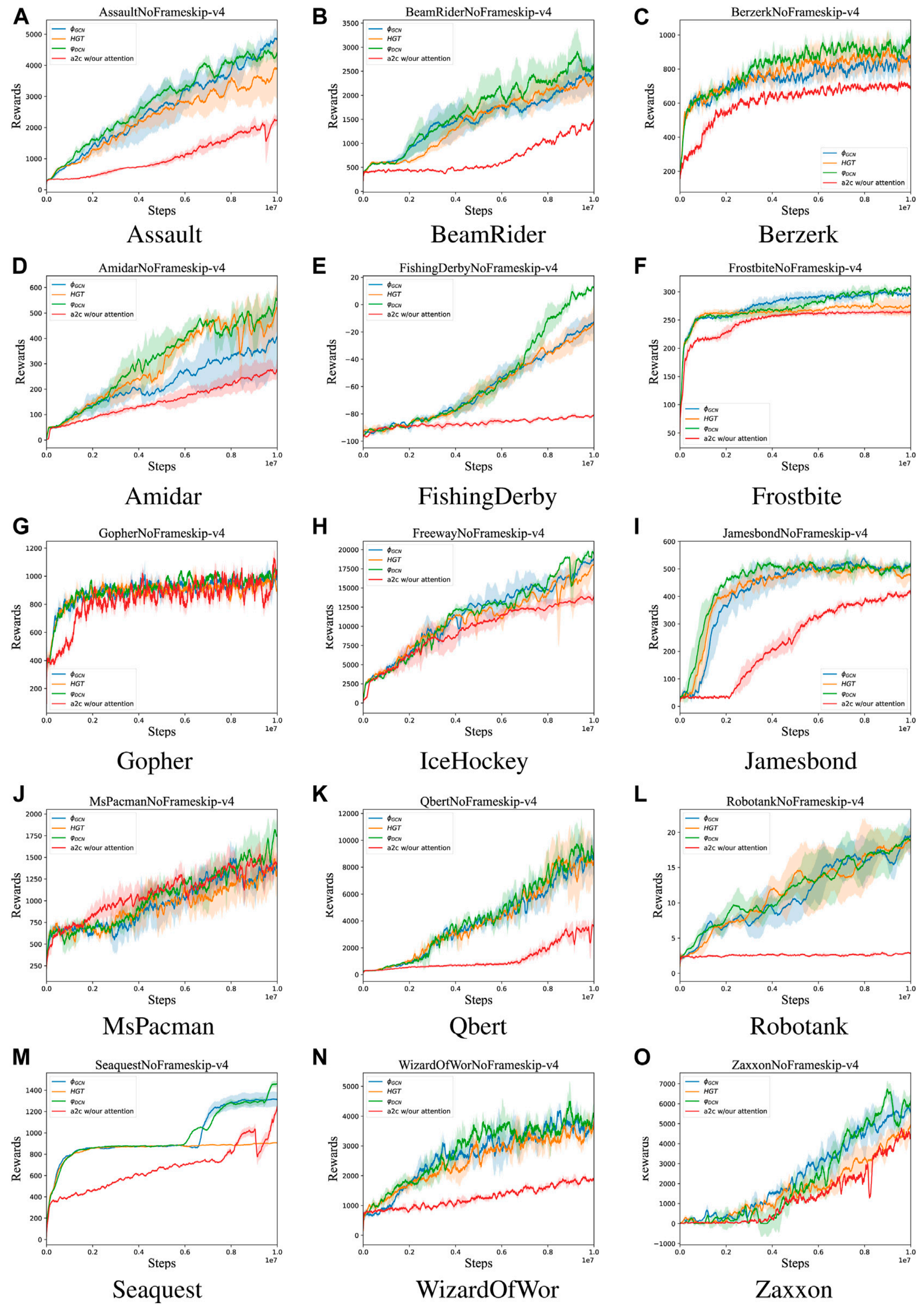

FIGURE 3
Comparison of rewards per episodes between a2c w/our attention, HGT, Φ_{GCN}, and φ_{DCN} on Atari. **(A)** Assault. **(B)** Beamrider. **(C)** Berzerk. **(D)** Amidar. **(E)** Fishing Derby. **(F)** Frostbite. **(G)** Gopher. **(H)**: IceHockey. **(I)** James Bond. **(J)** Ms. Pac-Man. **(K)** Q*bert. **(L)** Robo Tank. **(M)** SeaQuest. **(N)** Wizard Of Wor. **(O)** Zaxxon.

structures in different directions. In the MDP with temporal attributes, the undirected symmetric adjacency matrix A_u cannot be adopted. To approximate the transition matrix in the MDP, we use the digraph Laplacian. Given a directed graph (digraph) $G = (V_d, E_d)$, the adjacency matrix is expressed as $A_d = \{0, 1\}^{n \times n}$, where the number of nodes is denoted by n. $X \in R^{n \times c}$ denotes the node features,

TABLE 2 Mean reward for 10 training processes on Atari. The better result is bolded.

	Alien	Amidar	Assault
ϕ_{GCN}	1,385.2	406.2	**4,845.1**
φ_{DCN}	**1,423.6**	**549.5**	4,403.4
	Beamrider	Berzerk	Breakout
ϕ_{GCN}	2,357.9	818.7	150.2
φ_{DCN}	**2,631.1**	**973.3**	**164.2**
	Demon Attack	Fishing Derby	Frostbite
ϕ_{GCN}	9,807.9	−13.1	296.3
φ_{DCN}	**12,448.6**	**12.5**	**305.2**
	Gopher	IceHockey	James Bond
ϕ_{GCN}	1,010.4	**−5.4**	511.2
φ_{DCN}	**1,045.8**	−5.7	**519.2**
	Ms. Pac-Man	Q*bert	Robo Tank
ϕ_{GCN}	1,422.3	**8,968.0**	**19.4**
φ_{DCN}	**1,754.4**	8,551.5	18.9
	SeaQuest	Wizard Of Wor	Zaxxon
ϕ_{GCN}	1,310.4	3,670.0	5,591.2
φ_{DCN}	**1,459.1**	**4,054.2**	**6,134.0**

with c being the number of features. Using Eq. 10 to define the digraph Laplacian, which is symmetric, we can then derive the digraph convolution definition as follows:

$$Z_d = \frac{1}{2}\left(\Pi_{pr}^{\frac{1}{2}}\tilde{P}_{pr}\Pi_{pr}^{-\frac{1}{2}} + \Pi_{pr}^{-\frac{1}{2}}\tilde{P}_{pr}^{T}\Pi_{pr}^{\frac{1}{2}}\right)XW, \tag{11}$$

where $\tilde{P}_{pr}$ denotes a transition matrix with self-loops. Therefore, the message propagated by φ_{DCN} is as follows:

$$m_i = \mathrm{ReLU}\left(W^T\sum_{j}\frac{1}{2}\left(\Pi_{pr}^{\frac{1}{2}}\tilde{P}_{pr}\Pi_{pr}^{-\frac{1}{2}} + \Pi_{pr}^{-\frac{1}{2}}\tilde{P}_{pr}^{T}\Pi_{pr}^{\frac{1}{2}}\right)_{ij} m_j\right), \tag{12}$$

where the node j is a neighbor of node i and m_j is a message from node j.

Throughout φ_{DCN}, each state is represented by a node, while edges represent the transition probability between these states. Information about rewarding states is propagated through the message-passing mechanism of a directed graph convolutional network. In this paper, we propose a two-layer network as follows:

$$\begin{aligned}\varphi_{DCN} = \mathrm{softmax}\Bigg(&\frac{1}{2}\left(\Pi_{pr}^{\frac{1}{2}}\tilde{P}_{pr}\Pi_{pr}^{-\frac{1}{2}} + \Pi_{pr}^{-\frac{1}{2}}\tilde{P}_{pr}^{T}\Pi_{pr}^{\frac{1}{2}}\right)\\ &\mathrm{ReLU}\left(\frac{1}{2}\left(\Pi_{pr}^{\frac{1}{2}}\tilde{P}_{pr}\Pi_{pr}^{-\frac{1}{2}} + \Pi_{pr}^{-\frac{1}{2}}\tilde{P}_{pr}^{T}\Pi_{pr}^{\frac{1}{2}}\right)XW^{(0)}\right)W^{(1)}\Bigg).\end{aligned} \tag{13}$$

Then, we can express the loss function ℓ of φ_{DCN} as follows:

$$\ell = \ell_0 + \eta\ell_{prop}, \tag{14}$$

where the parameter η is the weight assigned to the propagation loss ℓ_{prop}. Here, the supervised loss ℓ_0 is defined as the cross-entropy between the prediction result $\hat{Y}$ and the ground-truth label Y, denoted by the symbol $H(Y,\hat{Y})$. Y represents the probability that the path taken at the moment is the optimal trajectory. It is worth mentioning that $\hat{Y}$, found at the output of φ_{DCN}, is defined as a probability distribution $\varphi_{DCN}(S)$. According to the results of this study, we have calculated the supervised loss ℓ_0 as follows:

$$\ell_0 = H\left(P(O\,|\,S), \varphi_{DCN}(S)\right) = \sum_S P(O\,|\,S)\log\left(\varphi_{DCN}(S)\right). \tag{15}$$

A propagation loss implemented as a recursive case is identified as ℓ_{prop} in Eq. 14. The recursive case of the message-passing mechanism can be implemented by the propagation loss ℓ_{prop} as follows:

$$\ell_{prop} = \sum_{vi,vj}\tilde{A}_{d_{v_i,v_j}}\left\|\varphi_{DCN}\left(X_{v_i}\right) - \varphi_{DCN}\left(X_{v_j}\right)\right\|^2. \tag{16}$$

4.3 Training

This paragraph describes the training process of φ_{DCN}. We propagate information about rewarding states through the message-passing mechanism of this directed graph convolution neural network. Then, the potential function $\Phi_{\alpha\beta}(\cdot)$ is learned on propagated messages $\alpha(S_t, A_t)$ and $\beta(S_t, A_t)$ (as in Eq. 9). Once the potential function $\Phi_{\alpha\beta}(\cdot)$ is learned, the new reward function R (S_t, A_t, S_{t+1}) is calculated to accelerate RL by replacing the original reward function r (S_t, A_t).

In this case, the combined value function Q^{π}_{comb} of RL can be calculated using the sum of two value functions $Q^{\pi}_{comb}(s,a) = \delta Q^{\pi}(s,a) + (1-\delta)Q^{\pi}_{\varphi}(s,a)$, where $Q^{\pi}(s,a) = \mathrm{E}\left[\sum_t \gamma^t r(S_t, A_t)\right]$ is the original Q-value function and $Q^{\pi}_{\varphi}(s,a) = \mathrm{E}\left[\sum_t \gamma^t r(S_t, A_t) + \gamma\varphi_{DCN}(S_{t+1}) - \varphi_{DCN}(S_t)\right]$ is the reward-shaped function. Two value functions can be weighted by the parameter δ. In this paper, we execute reward shaping φ_{DCN} on the underlying method PPO [31], which is a policy-based approach. The training process of φ_{DCN} is described in Algorithm 1.

```
Create an empty digraph $G$
for $Episode = 0, 1, 2, \ldots$ do
  while $t < T$ do
    Add transition $(S_t, S_{t+1})$ to digraph $G$
  end while
  if mod$(Episode, N)$ then
    Train $\varphi_{DCN}$ on the digraph $G$
  end if
  $Q^{\pi}_{comb} = \mu Q^{\pi} + (1-\mu)Q^{\pi}_{\Phi}$
  Maximize $E_{\pi}[\nabla \log \pi(A_t\,|\,S_t) Q^{\pi}_{comb}(S_t, A_t)]$
end for
```

Algorithm 1. Directed graph convolution neural networks for reward shaping.

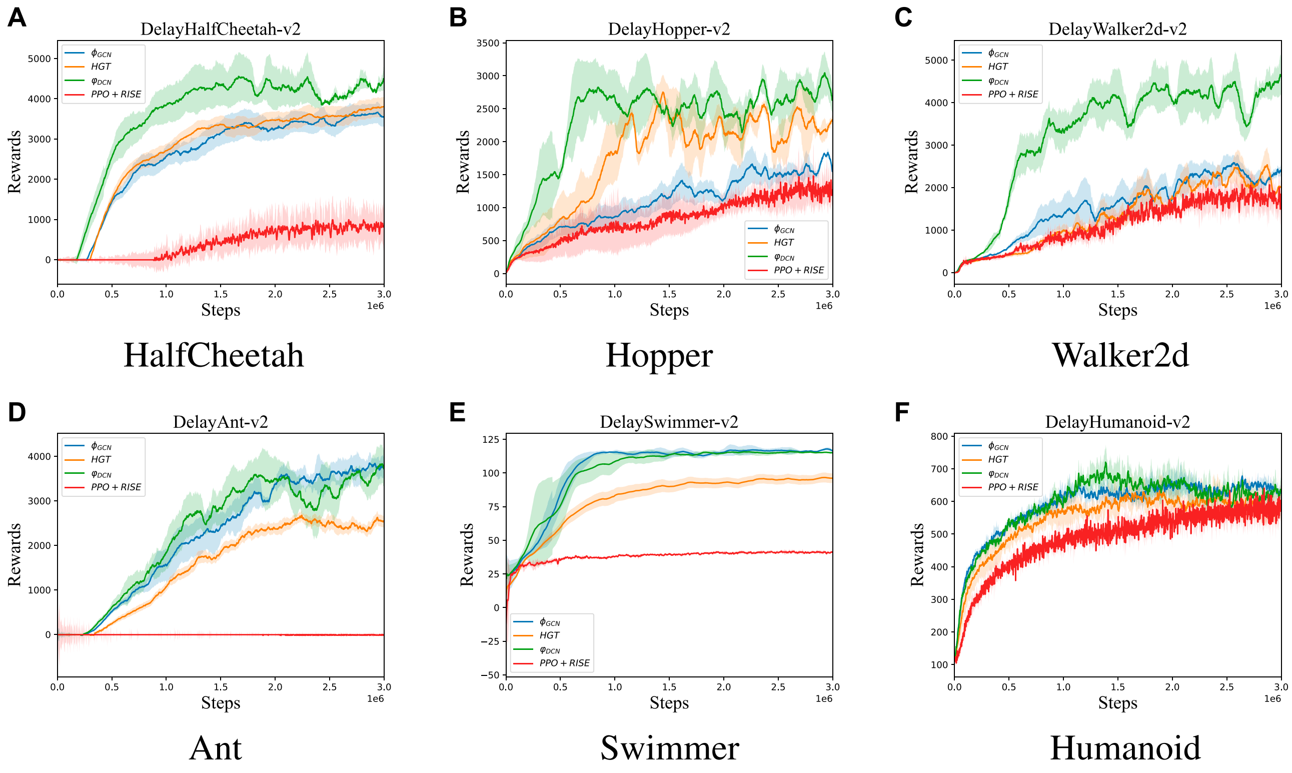

FIGURE 4
Delay rewards. Comparison of rewards per episode between HGT, PPO + RISE, Φ_{GCN}, and φ_{DCN} on MuJoCo. **(A)** HalfCheetah. **(B)** Hopper. **(C)** Walker2D. **(D)** Ant. **(E)** Swimmer. **(F)** Humanoid.

TABLE 3 Mean reward for 10 training processes on MuJoCo. The better result is bolded.

	HalfCheetah	Hopper	Walker2D
ϕ_{GCN}	3,543.9	1,550.4	2,413.5
φ_{DCN}	**4,496.8**	**2,630.6**	**4,620.9**
	Ant	Swimmer	Humanoid
ϕ_{GCN}	3,660.1	**116.6**	625.7
φ_{DCN}	**3,722.5**	115.0	**633.3**

5 Experiment

5.1 Experimental setup

In the experiment, the benchmarks from Atari and MuJoCo are adopted for evaluation.

Atari plays an important role in the field of RL [15]. Atari includes many classic games, such as Pac-Man, Space Invaders, Asteroids, and Pitfall. These games have become indelible symbols in the history of electronic games. Atari games are widely used as test platforms and benchmarks in the field of RL. Atari games have diversity, complexity, and challenges, covering various types of games, such as shooting, action, and adventure, at various difficulty levels. Therefore, they are used to evaluate the performance of the RL algorithm on complex tasks and help researchers understand the advantages and limitations of the algorithm.

MuJoCo is a physics engine and simulator, which provides researchers and developers with an efficient and accurate physical simulation environment for training and evaluating RL algorithms [16]. MuJoCo provides a high-performance physics simulation engine that can simulate the dynamics and physical interactions of complex multi-joint robots and objects. This allows researchers to quickly test and verify RL algorithms in a simulation environment without the need for actual robots or real environments. MuJoCo supports various types of tasks and environmental settings, including robot control, object grasping, and movement [32]; [33,34]. It also allows users to customize the environment as needed to meet various research and application needs.

For the two categories of games, Atari and Mujoco, Table 1 shows the hyperparameters in the φ_{DCN}. The hardware components of our system include an RTX2070 GPU, CPU E5-2670V3, and 32 GB RAM.

We set ϕ_{GCN} as the baseline algorithm for comparison. In the literature [13], ϕ_{GCN} has been experimentally demonstrated to have better performance than others, such as the PPO [31], RND [35], ICM [36], and LIRPG [37]. For the purpose of comparing the proposed approach φ_{DCN} directly with other latest algorithms, PPO + RISE [38], a2c w/our attention [39], and HGT [29] are adopted as contenders. For a fair comparison, all competing algorithms use the default hyperparameters.

5.2 Experimental results on Atari

Due to its reactive and hard-exploration nature, the Atari benchmark is used for experiments. We repeat the experiment 10 times over ten million frames from each game in order to assess the applicability and effectiveness of the proposed φ_{DCN}.

In this experiment, we use potential-based reward shaping approaches HGT and ϕ_{GCN} as comparison methods, where HGT is an extended version of ϕ_{GCN}. We notice that HGT mines the logical correlations between states by enriching the propagated messages. In addition, we also compare a2c w/our attention as it is designed to improve exploration ability. However, a2c w/our attention does not guarantee the invariance of the optimal policy.

Figure 3 presents the mean rewards obtained from the 10 training processes using Atari tasks. In accordance with this, the proposed φ_{DCN} demonstrated good improvements in most games, including Ms Pac-Man, which displayed a 23% higher reward than the baseline ϕ_{GCN}. It is also observed that similar results are observed for Gopher, Demon Attack, and Amidar. Based on the results given in Table 2, the φ_{DCN} approach performs better than all other games in terms of improving learning performance by an average of 12.3%. It is concluded that reward shaping is enhanced by the message-passing mechanism in directed graph convolutional networks. A further analysis is conducted, with the results given in Table 2, which demonstrates empirically that the use of a directed graph Laplacian leads to performance improvement.

5.3 Experimental results on MuJoCo

In this experiment, we evaluate the performance of φ_{DCN} in continuous control tasks. We considered six standard environments in MuJoCo, namely, Ant, Humanoid, Hopper, Swimmer, Walker2D, and HalfCheetah. In order to increase the difficulty of the experiment, we used an environment with a delayed reward version that makes reward sparse. In this setting, agents only receive accumulated rewards for a period of time (20 steps), rather than receiving rewards for each step. Here, we choose ϕ_{GCN}, PPO + RISE, and HGT for comparison, which are considered strong state-of-the-art baselines.

According to Figure 4, when φ_{DCN} is trained on delayed reward environments, it is generally faster than baselines in all six MuJoCo environments. Particularly, φ_{DCN} achieves significant performance improvement in delayed reward environmental HalfCheetah, Hopper, and Walker2D. Although our approach φ_{DCN} performs at a similar level to ϕ_{GCN} which executes reward shaping through the message-passing mechanism on undirected graphs, we have surpassed the other two algorithms, namely, HGT and PPO + RISE. It is evidenced from Table 3 that our proposed approach has achieved much better rewards (37.1% higher) than the baseline ϕ_{GCN}. In this study, it is evident that the performance of the φ_{DCN} is improved in continuous control tasks. This suggests that the proposed approach holds promise for accelerating RL in continuous control tasks when rewards are sparse.

5.4 Ablation analysis

An ablation analysis is conducted to determine how the directed graph Laplacian affects performance, as illustrated in Table 2 and Table 3. It should be noted that the bold one is the better one. The only difference between φ_{DCN} and ϕ_{GCN} is the graph Laplacian, where φ_{DCN} is the directed graph Laplacian and ϕ_{GCN} is the undirected graph Laplacian. According to this study, directed graph convolution networks have significantly improved performance in most of environments. Message passing with directional attributes can improve its performance. There is an improvement of 12.3% in the Atari experiment as compared to ϕ_{GCN}. The generalizability of φ_{DCN} is also demonstrated in several Atari tasks. Particularly in the continuous control tasks (MuJoCo), the performance is improved by an average of 37.1%.

6 Conclusion

Game theory utilizes the concepts and methods of RL to solve decision-making problems. However, the challenge of sparse rewards often exists in RL. Our proposed approach φ_{DCN} has been shown to be more effective in this issue as the message-passing mechanism of the directed graph Laplacian can be utilized to accelerate RL. In preliminary experiments conducted on Atari and MuJoCo, the proposed φ_{DCN} has demonstrated substantial improvement over conventional graph convolutional networks with an impressive increase of 12.3% and 37.1% compared to competing algorithms in terms of rewards per episode.

Despite this, there are still some shortcomings in certain aspects of φ_{DCN}, such as the high computational overhead of directed graph convolution operations. We are planning to conduct further research on this issue as the primary focus of our next project.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary Material; further inquiries can be directed to the corresponding authors.

Author contributions

JS: conceptualization, data curation, investigation, methodology, software, validation, and writing–original draft. ZA: writing–review and editing. HY: data curation, funding acquisition, project administration, resources, and writing–review and editing. YW: writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Zhejiang Science and Technology Major Program on Agricultural New Variety Breeding (2021C02070-1) and the Nonprofit Research Projects (CAFYBB2022XC001-2) of Chinese Academy of Forestry.

Acknowledgments

The authors would like to thank the editors and reviewers for their efforts in supporting the publication of this paper.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors, and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Song W, Sheng W, Li D, Wu C, Ma J. Modeling complex networks based on deep reinforcement learning. *Front Phys* (2022) 9:836. doi:10.3389/fphy.2021.822581

2. Grech L, Valentino G, Alves D, Hirlaender S. Application of reinforcement learning in the lhc tune feedback. *Front Phys* (2022) 10:929064. doi:10.3389/fphy.2022.929064

3. Guo H, Wang Z, Song Z, Yuan Y, Deng X, Li X. Effect of state transition triggered by reinforcement learning in evolutionary prisoner's dilemma game. *Neurocomputing* (2022) 511:187–97. doi:10.1016/j.neucom.2022.08.023

4. Ladosz P, Weng L, Kim M, Oh H. Exploration in deep reinforcement learning: a survey. *Inf Fusion* (2022) 85:1–22. doi:10.1016/j.inffus.2022.03.003

5. Han D, He Y. The reinforcement learning model with heterogeneous learning rate in activity-driven networks. *Int J Mod Phys C* (2023) 34:2350092. doi:10.1142/s0129183123500924

6. Han D, Li X. On evolutionary vaccination game in activity-driven networks. *IEEE Trans Comput Soc Syst* (2022) 10:142–52. doi:10.1109/tcss.2021.3137724

7. Meyn S. *Control systems and reinforcement learning*. Cambridge University Press (2022).

8. Zhu Y, Pang J-H, Tian F-B. Point-to-point navigation of a fish-like swimmer in a vortical flow with deep reinforcement learning. *Front Phys* (2022) 10:870273. doi:10.3389/fphy.2022.870273

9. Chen L, Wang C, Zeng C, Wang L, Liu H, Chen J. A novel method of heterogeneous combat network disintegration based on deep reinforcement learning. *Front Phys* (2022) 10:1021245. doi:10.3389/fphy.2022.1021245

10. Sang J, Wang Y. Graph convolution with topology refinement for automatic reinforcement learning. *Neurocomputing* (2023) 554:126621. doi:10.1016/j.neucom.2023.126621

11. Lu R, Jiang Z, Wu H, Ding Y, Wang D, Zhang H-T. Reward shaping-based actor–critic deep reinforcement learning for residential energy management. *IEEE Trans Ind Inform* (2022) 19:2662–73. doi:10.1109/tii.2022.3183802

12. Ng AY, Harada D, Russell S. Policy invariance under reward transformations: theory and application to reward shaping. Icml *(Citeseer)* (1999) 99:278–87.

13. Klissarov M, Precup D. *Reward propagation using graph convolutional networks* (2020). *arXiv preprint arXiv:2010.02474*.

14. Sami H, Bentahar J, Mourad A, Otrok H, Damiani E. Graph convolutional recurrent networks for reward shaping in reinforcement learning. *Inf Sci* (2022) 608:63–80. doi:10.1016/j.ins.2022.06.050

15. Bellemare MG, Naddaf Y, Veness J, Bowling M. The arcade learning environment: an evaluation platform for general agents. *J Artif Intelligence Res* (2013) 47:253–79. doi:10.1613/jair.3912

16. Todorov E, Erez T, Tassa Y. Mujoco: a physics engine for model-based control. In: *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE (2012). p. 5026–33.

17. Harutyunyan A, Devlin S, Vrancx P, Nowé A Expressing arbitrary reward functions as potential-based advice. In: *Proceedings of the AAAI conference on artificial intelligence* (2015), 29.

18. Xiao B, Ramasubramanian B, Clark A, Hajishirzi H, Bushnell L, Poovendran R (2019). Potential-based advice for stochastic policy learning. In *2019 IEEE 58th conference on decision and control (CDC)* (IEEE), 1842–9.

19. Devlin SM, Kudenko D. Dynamic potential-based reward shaping. In: *Proceedings of the 11th international conference on autonomous agents and multiagent systems*. IFAAMAS (2012). 433–40.

20. Laud AD. *Theory and application of reward shaping in reinforcement learning*. University of Illinois at Urbana-Champaign (2004).

21. Tong Z, Liang Y, Sun C, Rosenblum DS, Lim A. *Directed graph convolutional network* (2020). *arXiv preprint arXiv:2004.13970*.

22. Li Y, Tarlow D, Brockschmidt M, Zemel R. *Gated graph sequence neural networks* (2015). *arXiv preprint arXiv:1511.05493*.

23. Li Q, Han Z, Wu X-M Deeper insights into graph convolutional networks for semi-supervised learning. In: *Proceedings of the AAAI conference on artificial intelligence* (2018), 32.

24. Kipf TN, Welling M. *Semi-supervised classification with graph convolutional networks* (2016). *arXiv preprint arXiv:1609.02907*.

25. Monti F, Otness K, Bronstein MM. Motifnet: a motif-based graph convolutional network for directed graphs. In *2018 IEEE data science workshop (DSW)*. (IEEE) (2018), 225–8.

26. Ma Y, Hao J, Yang Y, Li H, Jin J, Chen G. *Spectral-based graph convolutional network for directed graphs* (2019). *arXiv preprint arXiv:1907.08990*.

27. Ziebart BD, Maas AL, Bagnell JA, Dey AK. Maximum entropy inverse reinforcement learning. Aaai *(Chicago, IL, USA)* (2008) 8:1433–8.

28. Toussaint M, Storkey A. Probabilistic inference for solving discrete and continuous state markov decision processes. *Proc 23rd Int Conf Machine Learn* (2006) 945–52. doi:10.1145/1143844.1143963

29. Sang J, Wang Y, Ding W, Ahmadkhan Z, Xu L. *Reward shaping with hierarchical graph topology*. Pattern Recognition (2023), 109746.

30. Barker G, Schneider H. Algebraic perron-frobenius theory. *Linear Algebra its Appl* (1975) 11:219–33. doi:10.1016/0024-3795(75)90022-1

31. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. *Proximal policy optimization algorithms* (2017). *arXiv preprint arXiv:1707.06347*.

32. Hu P, Chu X, Lv L, Zuo K, Ni T, Wang T, et al. An efficient and secure data collection scheme for predictive maintenance of vehicles. *Ad Hoc Networks* (2023) 146:103157. doi:10.1016/j.adhoc.2023.103157

33. Zhao R, Wang Y, Xiao G, Liu C, Hu P, Li H. A method of path planning for unmanned aerial vehicle based on the hybrid of selfish herd optimizer and particle swarm optimizer. *Appl Intelligence* (2022) 52:16775–98. doi:10.1007/s10489-021-02353-y

34. Zhao R, Wang Y, Xiao G, Liu C, Hu P, Li H. A selfish herd optimization algorithm based on the simplex method for clustering analysis. *The J Supercomputing* (2021) 77:8840–910. doi:10.1007/s11227-020-03597-0

35. Burda Y, Edwards H, Storkey A, Klimov O. *Exploration by random network distillation* (2018). *arXiv preprint arXiv:1810.12894*.

36. Pathak D, Agrawal P, Efros AA, Darrell T. Curiosity-driven exploration by self-supervised prediction. In: *International conference on machine learning*. PMLR (2017). p. 2778–87.

37. Zheng Z, Oh J, Singh S. On learning intrinsic rewards for policy gradient methods. *Adv Neural Inf Process Syst* (2018) 31.

38. Yuan M, Pun M-O, Wang D. Rényi state entropy maximization for exploration acceleration in reinforcement learning. *IEEE Trans Artif Intelligence* (2022) 4:1154–64. doi:10.1109/tai.2022.3185180

39. Wu H, Khetarpal K, Precup D. Self-supervised attention-aware reinforcement learning. *Proc AAAI Conf Artif Intelligence* (2021) 35:10311–9. doi:10.1609/aaai.v35i12.17235