Multi-objective RL with Preference Exploration

Wei Xi and Xian Guo(✉)

Institute of Robotics and Automatic Information System, Nankai University, Tianjing 300353, China
guoxian@nankai.edu.cn

Abstract. Traditional multi-objective reinforcement learning problems pay attention to the expected return of each objective under different preferences. However, the difference in strategy in practice is also important. This paper proposes an algorithm Multi-objective RL with Preference Exploration (MoPE), which can cover the optimal solutions under different objective preferences as much as possible with only one trained model. Specifically, the coverage of the optimal solution is improved by exploring the preference space in the sampling stage and reusing samples with similar preferences in the training stage. Furthermore, for different preference inputs, a variety of diversity strategies that conform to the preference can be generated by maximizing the mutual information of preference and state based on a method of information theory. Compared with the existing methods, our algorithm can implement more diverse strategies on the premise of ensuring the coverage of the optimal solution.

Keywords: Multi-objective · Reinforcement learning · Diversity · Information theory

1 Introduction

In the real world, a task often contains multiple objectives, and the preferences for these objectives are different in different situations, so various strategies are needed to meet the needs in different situations. Existing reinforcement learning methods such as DQN [1] and DDPG [2] can already handle single-objective problems. In more cases, the importance of each objective in the task is difficult to determine in advance, or some changes need to be made according to the actual situation. In single-objective reinforcement learning, this kind of problem is solved by adjusting the weight of this objective separately. The results of this adjustment method are often uncertain, because some objective may not be independent of each other, or even exist in opposition to each other. And for the

This work is supported by the National Natural Science Foundation of China (62073176). All the authors are with the Institute of Robotics and Automatic Information System, College of Artificial Intelligence, Nankai University, China.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
H. Liu et al. (Eds.): ICIRA 2022, LNAI 13455, pp. 669–680, 2022.
https://doi.org/10.1007/978-3-031-13844-7_62

traditional reinforcement learning algorithm, its goal is to maximize the expected return made up of multiple parts, the larger return is not necessarily achieved by increasing the reward, but by reducing the penalty. For example, when the reward for reaching the goal is small, and the penalty related to time is large, the robot may end the episode early by colliding with the obstacle to reduce the penalty obtained. Compared with traditional single-objective reinforcement learning, multi-objective reinforcement learning (MORL) provides a better way to deal with multi-objective tasks.

Existing MORL methods can be divided into two categories [3]: outer loop and inner loop [5]. The outer loop method treats the MORL as multiple single-objective reinforcement learning problems with different preferences. This method achieves the effect of approximately representing the Pareto front by maintaining a population of policies [6] distinguished by preference, which makes it difficult to extend to complex problems or problems with a large number of objective. The inner loop class method learns a value function or policy network conditioned on preference, and uses the deep neural network to obtain an approximate Pareto optimal solution. During the training process, different preferences are used as the input of the neural network, and the goal is to maximize the cumulative return under the linear weighting of the preferences. Compared with the outer loop method, it avoids maintaining a large set of strategies, but there are problems of large sample demand and catastrophic forgetting. Friedman et al. [7] combined the MORL method with Hindsight Experience Replay (HER) [8] and Deep Deterministic Policy Gradients to achieve higher sample utilization in continuous action space. Abels et al. [9] implemented a weight-conditioned network (CN) based on vector Q-functions, and accelerate learning by using deverse experience replay (DER). Yang et al. [10] proposed the optimality operator for a generalized version of Bellman equation and the Envelope Q-learning (EQL) method, they proved its theoretical onvergence under linear preference. The EQL outperforms single-objective reinforcement learning on the complex Atari game SuperMario.

Multi-objective problems can be seen as a form of diversity problems. For multi-objective problems, the strategies on the Pareto frontier have certain differences. Diversity problems focus on differences in strategies under the same scenario, which can be caused by different reward functions. Shen et al. [6] used multi-objective genetic algorithm combined with single-objective reinforcement learning method to obtain different styles of game AI. Optimizing the distance of trajectories obtained by different strategies or the kl-distance of different strategies is also a way to generate diversity. Wu et al. [11] used the Maximum Mean Discrepancy (MMD) distance between trajectories as a regular term to generate different opponents strategies. Eysenbach et al. [12] used unsupervised learning for generating diversity by maximizing the mutual information of different skills and states based on maximum entropy reinforcement learning. These methods can generate more fine-grained diversity, but for many practical problems, a controllable and interpretable diversity is more valuable.

The contribution of our work is threefold:

1) We propose an algorithm which can cover the optimal solution as much as possible with training only one model.
2) For different preference inputs, a variety of diversity strategies that conform to the preference can be generated.
3) Compared with single-objective reinforcement learning, our algorithm can reduce the possibility of falling into a local optimal solution, and can better converge to the optimal solution in sparse reward environments.

The structure of this paper is organized as follows: In Sect. 2, we introduce the relevant background of multi-objective reinforcement learning and some notations used in this paper. In Sect. 3, we analyze some problems with existing methods and propose a multi-objective reinforcement learning algorithm with preference exploration (MoPE). In Sect. 4, we introduce two multi-objective environments and validate our methods based on them. In Sect. 5, we summarize the contributions of this article and illustrate future research directions.

2 Background

The Multi-objective Reinforcement Learning (MORL) is a class of methods relative to traditional single-objective reinforcement learning. Its reward function $\boldsymbol{r}(s,a) = (r_1(s,a), ...r_m(s,a))$ is given in vector form, where m is the number of targets. The MORL is derived from multi-objective optimization problems, whose goal is to solve $\max f(x) = (f_1(x), ...f_m(x))^\top$. In most multi-objective problems, there are conflicts or incomparability between objectives, and the global optimal solution cannot be obtained, which leads to the improvement of one objective that tends to weaken other objectives. For a solution, if there is no other solution that is better than it on all objectives, it is called a Pareto solution under the problem. A set of Pareto optimal solutions is called the Pareto optimal set, and the surface formed by the optimal set in space is called the *Pareto front*.

The Markov Decision Process (MDP) is the basic form of reinforcement learning (RL). MDP can be represented by the tuple (S, A, P, γ, r), with state space S, action space A, action $a \in A$, transition probability $P(s' \mid s, a)$, discount factor γ, reward function $r(s,a)$. The goal of reinforcement learning is to learn the strategy $\pi : S \times A \to [0,1]$ to obtain the maximum cumulative return $G_t = \sum_t^\infty \gamma^t r(s,a)$, at this time the strategy is called the optimal strategy π^*. The solution method of the optimal strategy can be divided into two types: based on the value function $V(s) = E_\pi[G_t \mid S_t = s]$ and $Q(s,a) = E_\pi[G_t \mid S_t = s, A_t = a]$ and based on the policy. The DQN is a representation of a value function-based approach, which value function is updated as:

$$Q(s,a)_{i+1} \leftarrow Q(s,a)_i + \alpha \left[r + \gamma \max_{a' \in A} Q\left(s', a'\right)_i - Q(s,a)_i \right] \tag{1}$$

The PPO is a policy base algorithm, which loss function is:

$$J_{PPO}(\theta) = \sum_{t=1}^{T} \frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{old}\left(a_t \mid s_t\right)} \hat{A}_t - \lambda \mathrm{KL}\left[\pi_{\text{old}} \mid \pi_\theta\right] \\ \hat{A}_t = \hat{R}_t - V_\phi\left(s_t\right) \tag{2}$$

The Multi-objective Markov Decision Process (MOMDP) can be represented by the tuple $(S, A, P, \boldsymbol{r}, \Omega, f_\Omega)$ with vector reward function $\boldsymbol{r}(s, a, s')$, the space of preference Ω, preference function $f_\omega = \boldsymbol{\omega}^T \boldsymbol{r}(s, a)$, preference $\boldsymbol{\omega} \in \Omega$. When $\boldsymbol{\omega}$ is fixed, each $\boldsymbol{\omega}$ corresponds to a MDP process. All optimal solutions form the Pareto coverage set (PCS) $\mathcal{F}^* := \{\hat{\boldsymbol{r}} \mid \hat{\boldsymbol{r}} \geq \hat{\nabla}', \forall \hat{\boldsymbol{r}}'\}$, where the return $\hat{\boldsymbol{r}}_t = \sum_t^\infty \gamma^t \boldsymbol{r}_t(s, a)$. The optimal strategies corresponding to all optimal solutions constitute the *Pareto front* of this problem. For all possible $\boldsymbol{\omega} \in \Omega$ constitute a convex coverage set (CCS) [10] which is a subset of PCS:

$$\mathrm{CCS} := \left\{\hat{\boldsymbol{r}} \in \mathcal{F}^* \mid \exists \omega \in \Omega \text{ s.t. } \omega^\top \hat{\boldsymbol{r}} \geq \boldsymbol{\omega}^\top \hat{\boldsymbol{r}}', \forall \hat{\boldsymbol{r}}' \in \mathcal{F}^*\right\}. \tag{3}$$

Our goal is to learn a general value function $\mathcal{Q}(s, a, \boldsymbol{\omega}, \theta)$ that can generalize to the entire preference space, and the policy obtained by this value function can cover the CCS as much as possible.

3 Multi-objective RL with Preference Exploration

The algorithm we propose is based on two ideas. The first is the exploration and utilization of existing samples. For inner loop methods, it is necessary to sample trajectories under different preferences during the training process to learn a general value function, which reduce sample size for each preference. To address this issue, we expand and explore in existing samples through HER and prioritization experience replay (PER) [13] combined with information theory to increase sample utilization. The second is the exploration of preference space during sampling. Existing methods usually directly obtain the preference used in each episode through uniform sampling. While it's favorable to use the already trained preferences and corresponding result as priors to guide the selection of new preferences.

Exploring and Utilizing Existing Experience is the key to improving sample efficiency. The HER is a technique for dealing with the sparse reward goal-condition problem, which improves the utilization of samples by relabeling the goals of existing trajectories as goals that can be reached by the current strategy. Preference is also a goal in this paper. The preference will only affect the action of the agent, but not the dynamics of the environment. Therefore, A relabeling can be used to the experience to improve the utilization of the sample. For each experience in the batch $(s, \boldsymbol{\omega}_b, a, \boldsymbol{r}, s')$ we additionally sample N_ω preferences $\boldsymbol{\omega} = \{\omega_0, \omega_1, ..., \omega_{N_\omega} \mid \forall \omega, \|\omega_i - \omega_b\|_2 \leq \sigma_\omega\}$ as new goals. Different from the

EQL, it can be noticed that policies with similar preferences will be more similar under linear preferences, which can provide more information to help the learning of the current policy, so we will limit the newly sampled preferences to the vicinity σ_ω of the actual sampled ω_b which called Similar Preference Exploration. After relabeling, The EQL's loss is used to update the network. The loss function L includes two parts:

$$\begin{aligned} L^{\mathrm{A}}(\theta) &= \mathbb{E}_{s,a,\boldsymbol{\omega}}\left[\|\boldsymbol{y} - \boldsymbol{Q}(s,a,\boldsymbol{\omega};\theta)\|_2^2\right] \\ L^{\mathrm{B}}(\theta) &= \mathbb{E}_{s,a,\boldsymbol{\omega}}\left[\left|\boldsymbol{\omega}^\top \boldsymbol{y} - \boldsymbol{\omega}^\top \boldsymbol{Q}(s,a,\boldsymbol{\omega};\theta)\right|\right] \\ L &= L^{\mathrm{A}} + L^{\mathrm{B}} \end{aligned} \tag{4}$$

where $\boldsymbol{y} = \mathbb{E}_{s'}\left[\boldsymbol{r} + \gamma \arg_Q \max_{a,\boldsymbol{\omega}'} \boldsymbol{\omega}^\top \boldsymbol{Q}\left(s', a, \boldsymbol{\omega}'; \theta_k\right)\right]$. Compared to the original DQN objective, the EQL takes the largest Q value among the sampled actions and preferences simultaneously. L^{A} updates the vector Q function, L^{B} is the auxiliary loss, and the existence of L^{B} is to reduce the influence of the discrete solutions on the frontier on the optimization. The optimal solution corresponding to the preference is the same, which increases the difficulty of learning the value function.

The HER focuses on the utilization of data in the current batch, and PER is used to focus on the utilization of historical data. PER is generally based on the TD error of each experience. Based on the above analysis, it is more beneficial to consider a preference interval rather than the empirical TD error under a single preference in the MORL problem:

$$\delta = \frac{1}{N_\omega} \sum_{i=0}^{N_\omega} \boldsymbol{\omega_i}^\top \boldsymbol{y} - \boldsymbol{\omega_i}^\top \boldsymbol{Q}(s,a,\boldsymbol{\omega_i};\theta). \tag{5}$$

The TD error takes into account the error of the value function, reflecting how well the value function is learned. However, discrete optimal solutions will make strategies under multiple preferences correspond to the same optimal solution, which increases the possibility of strategies falling into local optimality. At the same time, the multi-objective reinforcement learning problem pays more attention to the results of different strategies. But the differences in results do not fully correspond to the differences in strategies. In this regard, we use the method of information theory to deal with this problem. Our method is based on an idea that a large difference in preference ω corresponds to a large difference in the state S reached by the agent. It should be able to better distinguish the agent's preferences under different state, that is, maximize the mutual information $I(S;\omega)$ between state and the preferences.

$$\begin{aligned} I(S;\omega) &= (\mathcal{H}[\omega] - \mathcal{H}[\omega \mid S]) \\ &= \mathbb{E}_{\omega\sim p(\omega), s\sim\pi(\omega)}[\log p(\omega \mid s)] - \mathbb{E}_{\omega\sim p(\omega)}[\log p(\omega)] \\ &\geq \mathbb{E}_{\omega\sim p(\omega), s\sim\pi(\omega)}\left[\log q_\phi(\omega \mid s) - \log p(\omega)\right] \end{aligned} \tag{6}$$

Among them, $p(\omega \mid s)$ is difficult to calculate directly, a discriminator network $q_\phi(\omega \mid s)$ is trained to approximate it. It can be proved that the approximated

solution can provide a variational lower bound on mutual information. $I(S;\omega)$ is optimized by add this priority to PER, the priority sampling probability is:

$$q(s_i) = \delta + \alpha(\log q_\phi(\omega \mid s) - \log p(\omega)). \tag{7}$$

The above formula can be understood that we pay more attention to the state that is easy to distinguish, and the value function has not been estimated accurately. This kind of state also corresponds to the frontier of exploration, and α is used to balance the ratio of the two parts. For some adversarial tasks, the addition of the discriminator also brings convenience to opponent modeling.

Exploring in Preference Space is key to speeding up training and mitigating catastrophic forgetting. To guarantee coverage of the optimal solution, continuously train with different preferences is needed. After many updates, the neural network may forget some of the policies it learned earlier. In order to reduce the impact of forgetting and take advantage of the learned policy, we need to update the sampling probability $p(\omega)$ during training to purposefully learn some preferences.

Specifically, we expect to give higher sampling probabilities to preferences that currently have few visits or incomplete training. For the problem of visit frequency, we discretize the preference space into N_I intervals $I_i \in \Omega, i \in [0, N_I]$. Put each used preference into the corresponding interval to get the count N_i. Combined with the counting-based exploration method in reinforcement learning [15], we set $p(\omega) \propto (N_i + 0.01)^{-1/2}, \omega \in I_i$.

4 Experiment

Deep-Sea Treasure (DST) [14] is a simple multi-objective 10×11 grid world environment, which is often used for the verification of MORL algorithms. Its Pareto frontier is obtainable and convex. In the DST environment, there is an agent and several treasures with scores, and the treasure near the bottom of the map have higher scores. At the beginning of the round, the agent is in the upper left corner of the map, and the algorithm needs to control the agent to move in four directions until it reach the treasure. The agent's goal is to obtain the highest possible score while taking the fewest steps. The two parts of the rewards are the reward r_{score} for reaching the treasure, and the penalty for each step $r_{steps} = -1$ (Fig. 1).

In order to evaluate the coverage of the algorithm to the optimal solution in the CCS, we introduce the evaluation metric Coverage Ratio (CR). The calculation of CR is based on the coincidence of the optimal solution set $\mathcal{P}$ and CCS found by the algorithm [10]:

$$CR = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}, \tag{8}$$

where $\text{precision} = |\mathcal{P} \cap \text{CCS}| / |\mathcal{P}|$, $\text{recall} = |\mathcal{P} \cap \text{CCS}| / |\text{CCS}|$, represent the proportion of coincident solutions in the two sets. In practical, we approximate

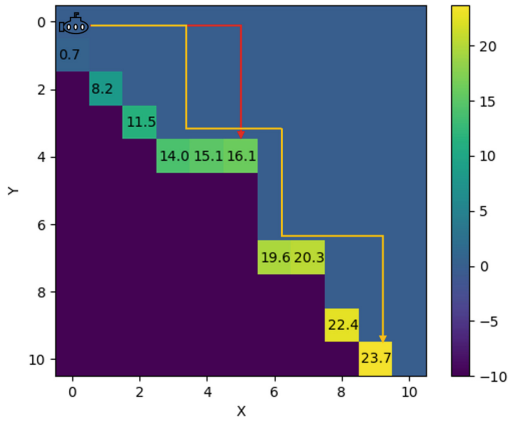
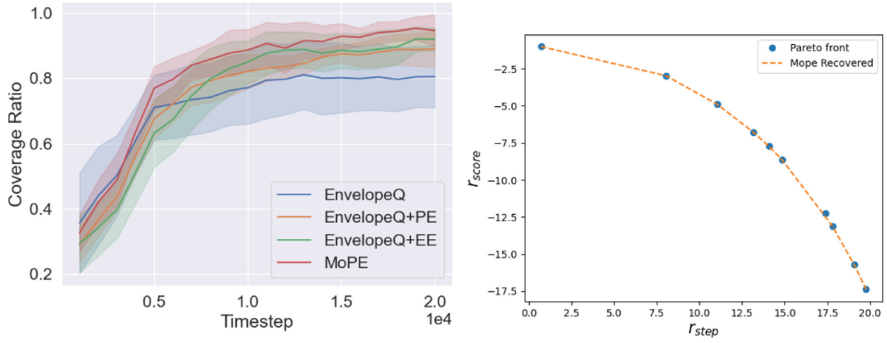


Fig. 1. The Deep-sea treasure (DST) environment. Agent starts from the upper left corner of the map, takes (x, y) as the state, and searches for treasures with different scores by moving in four directions. The red and the orange line in the figure represent two paths that can reach a specific treasure. It can be seen that the set of optimal solution paths in the DST environment is the shortest path to each treasure. (Color figure online)

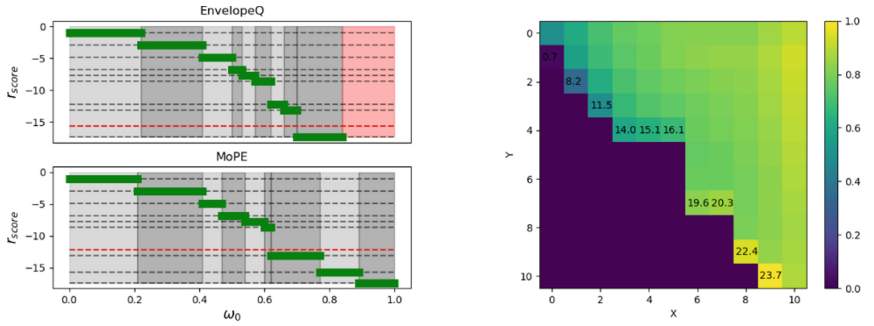


(a) (b)

Fig. 2. Evaluate Deep-sea treasure results. (a) Coverage ratio metric under 20000 timesteps. (b) The real CCS and the recovered solutions.

the current optimal solution set $\mathcal{P}$ by randomly sampling a certain number of ω in the preference space.

We use a conditional Multi-head Q network [9] by 4 fully connected hidden layers with {64, 128, 128, 64} for training. The inputs are states (x, y) and preferences, and the number of heads in the output layer is equal to the number of objectives. The original EQL and the algorithm after adding exploration were trained for 20,000 timesteps, and the CR was evaluated every 1,000 steps during the training process, and 2,000 preferences were evaluated using uniform sampling. For the experience exploration (EE) approach, a discriminator network



(a) (b)

Fig. 3. Deep-sea treasure results. (a) The relationship between preference and optimal solution under DST task. The gray dotted line is the Pareto optimal solution, the green thick solid line is the optimal solution obtained by the algorithm. The preference space can be divided into multiple parts according to the different solutions. The red dotted line indicates that the corresponding optimal solution is missing, and the red area indicates that the solution obtained by the algorithm is not in the optimal solution set. (b) The discriminator' s prediction of preference. The color indicates the weight corresponding to r_{score}. (Color figure online)

with 3 fully connected layers is used to output the probabilities used in PER. Then we sample 64 preferences under the condition of $\sigma_\omega = 0.04$ to relabel the samples. For the preference exploration (PE) approach, we divided the preference space into 10 parts. MoPE integrates these two methods of preference exploration. The final result is shown in Fig. 2. Under the same number of training steps, the algorithm after adding exploration can achieve higher CR than the original algorithm.

Compare the distribution of the optimal solutions obtained by the two algorithms in the preference space as shown in Fig. 3a. The solutions obtained by MoPE are more uniformly distributed in the preference space and are all within the CCS, while the EQL will have some suboptimal and concentrated solutions. Further, we plot the discriminator's prediction of preference as shown in Fig. 3b. For the DST environment, the optimal solution is unique, but the optimal path is not. The agent's preference is more difficult to discern when it is close to a treasure location in the top half of the map. Under our method, this part exhibits larger differences in predicted preferences, which means that agents choose more diversity routes for different preferences.

Robot Confrontation Game is a multi-objective problem in a continuous action space. There are two red and blue agents A_r, A_b in the scene, where A_r is the agent to be controlled, A_b is the rule-based agent, and the state update of the agents is based on (9).

$$\dot{x} = v\cos\theta, \dot{y} = v\sin\theta, \dot{\theta} = w, \dot{v} = a, \tag{9}$$

where robot's acceleration $a \in [-0.2, 0.2]$ m/s, angular velocity $w \in [-0.8, 0.8]$ rad/s, direction $\theta \in [-\pi, \pi]$. The goal is to hunt down the blue agent without going out of bounds. The definition of winning is that the distance and relative angle between the two agents are less than a certain threshold, such as (10) (Fig. 4).

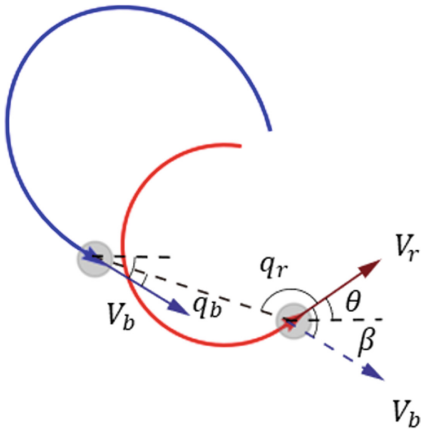


Fig. 4. Parameter definition of the environment.

$$q_r < 30°, q_b > 30°, \beta < 40°, d < 0.3\,m, \tag{10}$$

where q_r, q_b is the angle between the speed direction and the line connecting the two agents, β is the angle between the speed directions, and d is the distance between the two robots. For the strategy of rule-based agents, we use the threat index T which widely used in air combat problems as the evaluation index of the state, combined with single-step forward prediction, then select the action with the highest threat to the opponent. The threat index T we use includes two parts, the angle threat T_a and the distance threat T_b, which are defined as follows:

$$T_a = \frac{q_b - q_r}{\pi}, T_d = \frac{d_{\max} + d_{\min} - 2d}{d_{\max} - d_{\min}}, T = aT_a + bT_b, \tag{11}$$

where d_{max} and d_{min} are the maximum and minimum distances that two agents may encounter, a and b are the corresponding weight coefficients. In this paper, $a = 0.1, b = 0.9$.

For this environment we choose pursuit reward r_{catch} and moving distance reward r_{move} as optimization goals.

$$r_{catch} = \begin{cases} 10 & \text{if red wins} \\ -10 & \text{if bule wins} \\ (T_a - 1)/10 + (T_b - 1)/10 & \text{otherwise} \end{cases} \tag{12}$$

$$r_{run} = \begin{cases} -10 & \text{if out of range} \\ vdt & \text{otherwise} \end{cases}$$

In this environment, the capability of the two agents is consistent, which makes it easy for the algorithm to fall into a sub-optimal solution that keeps accelerating in circles. At this time, although the agent cannot obtain rewards, it will not be punished for being caught up. To alleviate this problem we use the threat index penalty to guide the agent.

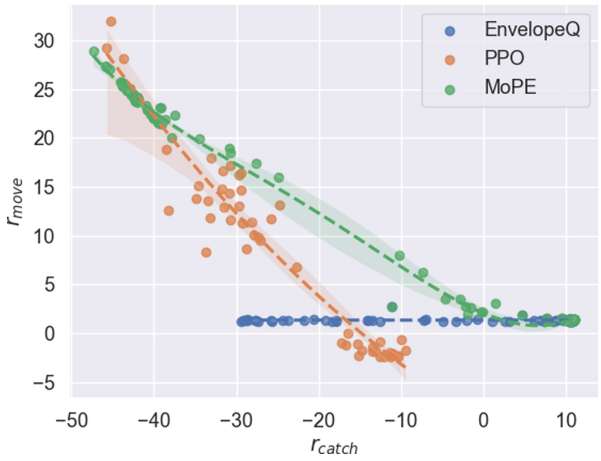


Fig. 5. Optimal solutions obtained by different algorithms. The dashed line is result fitted by data.

Compare the optimal solutions obtained by different algorithms in Fig. 5. The data of the PPO [16] is obtained by training single-objective problems under different preferences, and the data of MoPE and the EQL are obtained by uniform sampling 100 preferences after training. From the maximum reward that the algorithm can achieve, it can be seen that (1) the single-target PPO algorithm does not learn a strategy that can catch up with the opponent (the reward for catching up with the opponent is 10). (2) Compared with the EQL, MoPE has more diverse strategies. We further map MoPE's representative games under different preferences as Fig. 6.

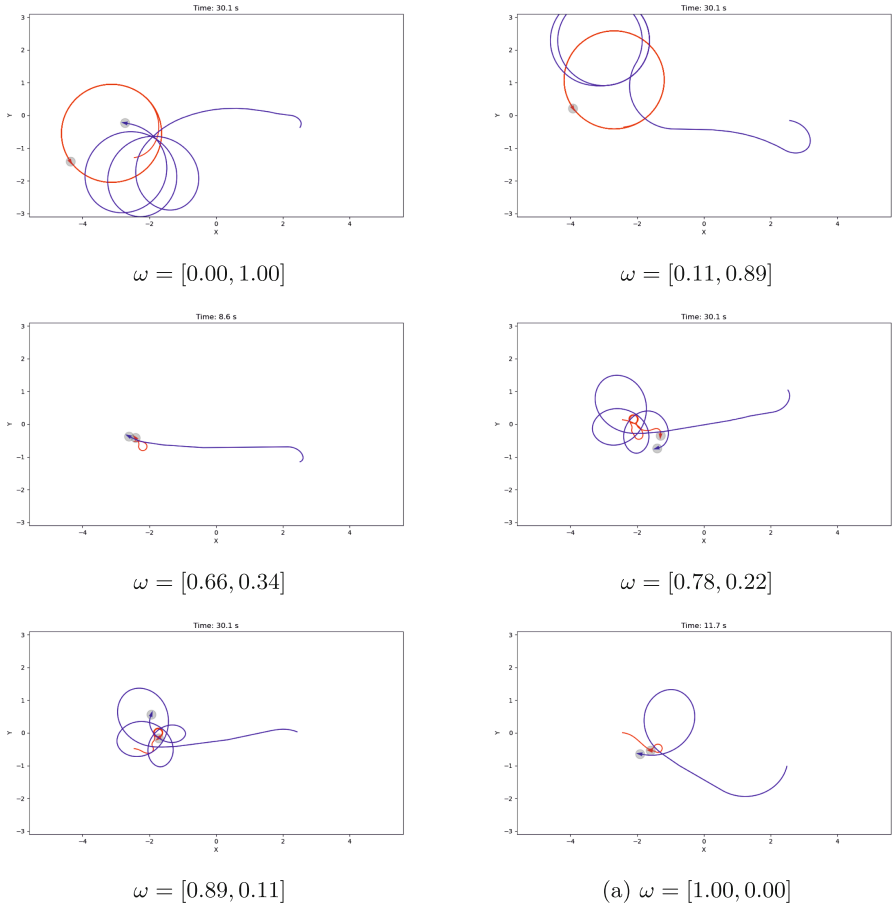


Fig. 6. MoPE result with different preferences.

5 Conclusion

We propose a multi-objective reinforcement learning algorithm (MORL), which can cover the optimal solutions under different preferences as much as possible when only one model is trained, and can generate sufficiently diversity strategies under different preference inputs. We conduct experiments on the algorithm in a simple grid world environment DST and a more difficult robot confrontation environment, Our experiments demonstrate that our algorithm has sufficient generalization and diversity relative to the benchmark algorithms. Future research will consider constrained multi-objective problems. The method in this paper is based on the assumption of linear preference, which means that all rewards must participate in the weighting calculation, and their corresponding weights need to be sampled in training, but for some penalty items, such as collisions, timeouts, etc., we do not need to consider them preferences, but wish

to constrain them within a certain range. One possible approach to this problem is Thresholded Lexicographic Ordering (TLO) [17].

References

1. Mnih, V., Kavukcuoglu, K., Silver, D., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
2. Lillicrap, T.P., et al.: Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2015)
3. Hayes, C.F., Rădulescu, R., Bargiacchi, E., et al.: A practical guide to multi-objective reinforcement learning and planning. arXiv preprint arXiv:2103.09568 (2021)
4. Czarnecki, W.M., Gidel, G., Tracey, B., et al.: Real world games look like spinning tops. Adv. Neural. Inf. Process. Syst. **33**, 17443–17454 (2020)
5. Roijers, D.M., Whiteson, S.: Multi-objective decision making. Synth. Lect. Artif. Intell. Mach. Learn. **11**(1), 1–129 (2017)
6. Shen, R., Zheng, Y., Hao, J., et al.: Generating behavior-diverse game AIs with evolutionary multi-objective deep reinforcement learning. In: IJCAI, pp. 3371–3377 (2020)
7. Friedman, E., Fontaine, F.: Generalizing across multi-objective reward functions in deep reinforcement learning. arXiv preprint arXiv:1809.06364 (2018)
8. Andrychowicz, M., Wolski, F., Ray, A., et al.: Hindsight experience replay. Adv. Neural Inf. Process. Syst. **30** (2017)
9. Abels, A., Roijers, D., Lenaerts, T., et al.: Dynamic weights in multi-objective deep reinforcement learning. In: International Conference on Machine Learning. PMLR, pp. 11–20 (2019)
10. Yang R, Sun X, Narasimhan K. A generalized algorithm for multi-objective reinforcement learning and policy adaptation[J]. Advances in Neural Information Processing Systems, 2019, 32
11. Wu, Z., Li, K., Zhao, E., et al.: L2e: learning to exploit your opponent. arXiv preprint arXiv:2102.09381 (2021)
12. Eysenbach, B., Gupta, A., Ibarz, J., et al.: Diversity is all you need: learning skills without a reward function. arXiv preprint arXiv:1802.06070 (2018)
13. Schaul, T., Quan, J., Antonoglou, I., et al.: Prioritized experience replay. arXiv preprint arXiv:1511.05952 (2015)
14. Vamplew, P., Yearwood, J., Dazeley, R., et al.: On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts. In: Australasian Joint Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, pp. 372–378 (2008)
15. Strehl, A.L., Littman, M.L.: An analysis of model-based interval estimation for Markov decision processes. J. Comput. Syst. Sci. **74**(8), 1309–1331 (2008)
16. Schulman, J., Wolski, F., Dhariwal, P., et al.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
17. Gábor, Z., Kalmár, Z., Szepesvári, C.: Multi-criteria reinforcement learning. In: ICML, vol. 98, pp. 197–205 (1998)