Distributional Pareto-Optimal Multi-Objective Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Multi-objective reinforcement learning (MORL) has been proposed to learn control policies over multiple competing objectives with each possible preference over returns. However, current MORL algorithms fail to account for distributional preferences over the multi-variate returns, which are particularly important in real-world scenarios such as autonomous driving. To address this issue, we extend the concept of Pareto-optimality in MORL into distributional Pareto-optimality, which captures the optimality of return distributions, rather than the expectations. Our proposed method, called Distributional Pareto-Optimal Multi-Objective Reinforcement Learning (DPMORL), is capable of learning distributional Pareto-optimal policies that balance multiple objectives while considering the return uncertainty. We evaluated our method on several benchmark problems and demonstrated its effectiveness in discovering distributional Pareto-optimal policies and satisfying diverse distributional preferences compared to existing MORL methods.

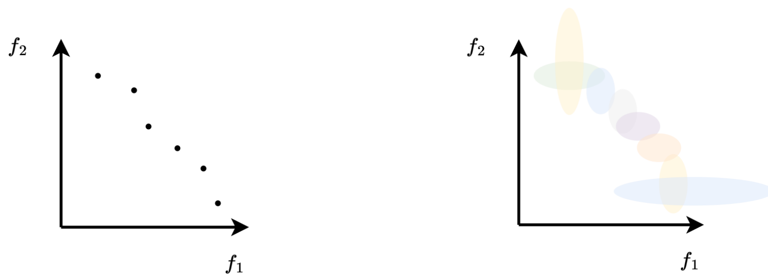
1 Introduction

Multi-Objective Reinforcement Learning (MORL) has recently received extensive attention in the artificial intelligence realm due to its adeptness at managing intricate decision-making issues with multiple conflicting objectives [1, 2, 3, 4]. In many multi-objective tasks, the relative preferences of users over different objectives are typically indeterminate a priori. Consequently, MORL's primary aim is to learn a variety of optimal policies under different preferences to approximate the Pareto frontier of optimal solutions. It has been demonstrated that MORL can significantly reduce the reliance on scalar reward design for objective combinations and dynamically adapt to the varying preferences of different users.

However, numerous real-world situations involve not only unknown relative preferences across multiple objectives, but also uncertain preferences in return distributions. These may include preference in risk [5, 6], safety conditions [7], and non-linear user's utility [8]. Consider, for instance, autonomous driving scenarios where agents need to strike a balance between safety and efficiency objectives. Different users might possess varying levels of risk tolerance. Some may demand high safety performance, tolerating lower expected efficiency, while others may seek a more balanced performance between safety and efficiency. Current MORL methods, by focusing exclusively on expected values with linear preferences, may not adequately capture multi-objective risk-sensitive preferences, hence being unable to deliver diverse policies catering to users with varied risk preferences.

In this work, we broaden the concept of Pareto-optimality in MORL to encompass distributional Pareto-optimality, which prioritizes the optimality of return distributions over mere expectations. From a theoretical perspective, we define Distributional Pareto-Optimal (DPO) policies, which capture the optimality of multivariate distributions through stochastic dominance [9, 10]. Distributional

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.



(a) MORL's Pareto-front

(b) Distributional MORL's Pareto-front

Figure 1: The comparison of learning targets between the traditional MORL tasks and distributional MORL tasks, in which f_1 and f_2 are two (conflicting) objectives.

Pareto-Optimality delineates the set of policies with optimal return distributions, which we formally establish as an extension of Pareto-optimality in MORL.

On the practical side, we propose a novel method named Distributional Pareto-Optimal Multi-Objective Reinforcement Learning (DPMORL). It aims to learn a set of DPO policies. We demonstrate that a policy achieving the highest expected utility for a given utility function is a DPO policy. We suggest an iterative process to learn diverse non-linear utility functions and optimize policies under them [11]. Experimental outcomes on several benchmark problems attest to DPMORL's effectiveness in optimizing policies that meet preferences on multivariate return distributions. We posit that our proposed method significantly addresses the challenge of managing multiple conflicting objectives with unknown preferences on multivariate return distributions in complex decision-making situations.

Our main contributions are listed as follows:

1. We introduce the concept of Distributional Pareto-Optimal (DPO) policies. This expands the notion of Pareto optimality in MORL to include preferences over the entire distribution of returns, not just their expected values. This enables agents to express more nuanced preferences over policies and align their decisions more closely with their actual objectives.
2. We propose DPMORL, a new algorithm for learning DPO policies under non-linear utility functions. This algorithm accommodates a broad range of distributional preferences, thus offering a more flexible and expressive approach to MORL.
3. We execute extensive experiments on various MORL tasks, demonstrating the effectiveness of our approach in learning DPO policies. Our findings show that our algorithm consistently surpasses existing MORL methods in terms of optimizing policies for multivariate expected and distributional preferences, underscoring its practical benefits.

2 Related Work

Multi-Objective Reinforcement Learning. MORL has emerged as a vibrant research area in the AI community owing to its adeptness in managing intricate decision-making scenarios involving multiple conflicting objectives [1, 2]. A plethora of MORL algorithms have been put forth in the literature. For instance, [12] proposed the utilization of Generalized Policy Improvement (GPI) to infer a set of policies, employing Optimistic Linear Support (OLS) for dynamic reward weight exploration [13]. This GPI-based learning process was further enhanced by [3] through the incorporation of a world model to augment sample efficiency. [4] proposed an evolutionary approach to learn policies with diverse weights. However, our proposed approach diverges from these by leveraging utility functions to guide policy learning [14, 15]. The utility-based paradigm, inspired by axiomatic approaches, accentuates the user's utility in decision-making problems, capitalizing on known information about the user's utility function and permissible policy types. Such methods scalarize the multi-dimensional objectives into a single reward function, enabling traditional RL algorithms to infer desirable policies, while also respecting axiomatic principles where necessary [2]. Several studies have utilized non-linear utility functions to guide policy learning, catering to more intricate preferences compared to linear ones [8, 16]. However, conventional MORL methodologies concentrate on optimizing the expected returns, neglecting the distributional characteristics of the returns. In contrast, our work extends the utility-based MORL framework by learning a set of utility functions with distributional attributes to guide policy learning.

Distributional Reinforcement Learning. Distributional RL extends traditional RL by modeling the entire distribution of returns, rather than just their expected values [17]. This approach has been shown to improve both learning efficiency and policy quality in a variety of single-objective RL tasks [18, 19]. Key algorithms in this area include Categorical DQN (C51) [17], Quantile Regression DQN (QR-DQN) [18], and Distributional MPO [19]. Despite the success of these distributional RL algorithms in single-objective settings, their direct extension to MORL has been limited due to the added complexity of handling multiple conflicting objectives and expressing preferences over the distribution of returns. On the other hand, the insights and techniques developed in distributional RL can provide a valuable foundation for incorporating distributional preferences into MORL, as demonstrated by our proposed approach.

Risk-Sensitive and Safe Reinforcement Learning. Risk-sensitive and safe RL are special cases of MORL that accentuate specific facets of reward distributions [2]. Risk-sensitive RL primarily considers reward variance, striving to optimize policies that negotiate the trade-off between expected return and risk [20, 21, 22]. Conversely, safe RL prioritizes constraints on the agent's behavior, ensuring adherence to specified safety criteria throughout the learning process [23, 24, 25]. Some studies have proposed the integration of constraints into the state space, constructing new Markov Decision Processes [7]. Others have explored the distributional aspects of rewards, investigating the implications of distributional RL on risk-sensitive and safety-oriented decision-making [18, 17]. However, our proposed setting is more general in terms of objectives, as it considers a broader range of user preferences and captures the entire distribution of rewards, rather than focusing solely on specific aspects such as risk, variance, or constraints. By extending MORL to incorporate distributional properties, our approach enables the learning of distributional Pareto-optimal policies that cater to diverse user preferences and offer better decision-making in a wide range of real-world applications.

3 Preliminaries: Multi-Objective Reinforcement Learning

In Multi-Objective Reinforcement Learning (MORL), the agent interacts with an environment modeled as a Multi-Objective Markov Decision Process (MOMDP) with multi-dimensional reward functions. A Multi-Objective MDP is defined by a tuple $(S, A, P, P_0, \boldsymbol{R}, \gamma, T)$, where S is the state space, A is the action space, P is the state transition function, $P_0(s)$ is the initial state distribution, $\boldsymbol{R} : S \times A \times S \to \mathbb{R}^K$ is a vectored reward function and K is the number of objectives, γ is the discount factor, T is the total timesteps[1]. The goal of the agent is to learn a set of Pareto-optimal policies, which represent the best possible trade-offs among the different objectives.

One popular methodology for MORL problems is the utility-based method, which combines the multi-dimensional reward functions into a single scalar reward function using a weighted sum or another aggregation method [26]. The intuition is to map the agent's preferences over different objectives to a scalar value for the agent training. Given a weight vector $\boldsymbol{w} = (w_1, w_2, \ldots, w_K)$, with w_i representing the importance of the i-th objective, the scalarized reward function is defined as $r_{\text{scalar}}(s, a) = \sum_{i=1}^{K} w_i r_i(s, a)$. The agent then solves the scalarized MDP by optimizing its policy to maximize the expected scalar reward, using standard reinforcement learning algorithms like Q-learning or policy gradient methods. This approach can be straightforward to implement and has been shown to be effective in various MORL settings under expected preferences [27]. However, such a linear combination of each dimension of the reward cannot deal with the preferences with distributional considerations.

4 Distributionally Pareto-Optimal Multi-Objective Reinforcement Learning

In this section, we introduce our proposed theoretical framework and algorithms for extending MORL to handle distributional preferences.

4.1 Distributionally Pareto-Optimal Policies

In this work, we consider the reward as a K-dimensional vector, where each element represents the reward for a specific objective. Given the MOMDP and a policy π, we define the random variable of

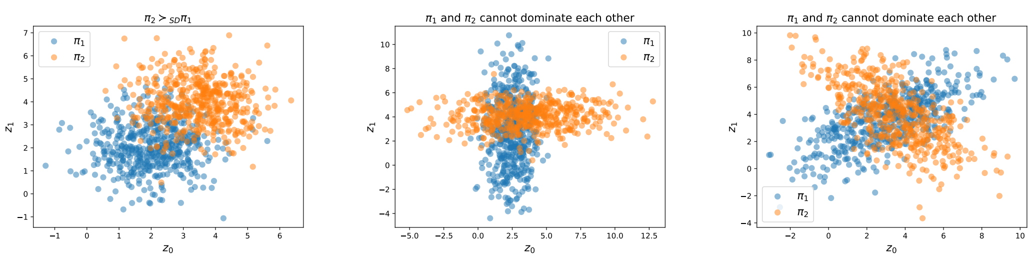
[1] We assume that current time t is a part of state to accommodate for the finite horizon setting.

multi-objective return as $\boldsymbol{Z}(\pi) = \sum_{t=0}^{T} \gamma^t \boldsymbol{r}_t$, where $\boldsymbol{r}_t$ is the multi-dimensional reward at step t, and the states $s_0, s_1, \cdots, s_T$ and actions $a_0, a_1, \cdots, a_T$ are sampled from the MOMDP and the policy π respectively. The return distribution of policy π, denoted as $\boldsymbol{\mu}(\pi)$, represents the joint distribution of the returns $\boldsymbol{Z}(\pi)$ following policy π. The utility function f is a non-decreasing function that maps a K-dimensional return into a scalar utility value, capturing the user's distributional preferences over the different objectives. The expected utility of policy π under the utility function f, represented as $\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}(\pi)} f(\boldsymbol{z})$, is the expected value of applying f to the return distribution $\boldsymbol{\mu}(\pi)$.

Our goal is to learn a set of policies that are distributionally Pareto-optimal, which means that their return distributions of each policy cannot dominate that of another. To measure such a relationship, here we first introduce the concept of stochastic dominance:

Definition 1 (Stochastic Dominance for Multivariate Distribution)**.** *A multivariate distribution $\boldsymbol{\mu}_1$ dominates another distribution $\boldsymbol{\mu}_2$, denoted as $\boldsymbol{\mu}_1 \succ_{SD} \boldsymbol{\mu}_2$, if and only if $\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2$ and for any non-decreasing utility function $f : \mathbb{R}^K \to \mathbb{R}$, $\boldsymbol{\mu}_1$ has greater expected utility than $\boldsymbol{\mu}_2$, i.e. $\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_1} f(\boldsymbol{z}) \geq \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_2} f(\boldsymbol{z})$.*

Definition 2 (Stochastic Dominance for Policies)**.** *A policy π_1 stochastically dominates another policy π_2, denoted as $\pi_1 \succ_{SD} \pi_2$, if and only if $\boldsymbol{\mu}(\pi_1) \succ_{SD} \boldsymbol{\mu}(\pi_2)$, indicating π_1 has greater expected utility than π_2 under any non-decreasing utility function f.*



(a) Stochastic Dominance Example 1 (b) Stochastic Dominance Example 2 (c) Stochastic Dominance Example 3

Figure 2: Three 2D examples of stochastic dominance given synthetic returns of two policies. (a) $\pi_2 \succ_{SD} \pi_1$, since any non-decreasing utility function satisfies $\mathbb{E}f(\mathbf{Z}(\pi_2)) \geq \mathbb{E}f(\mathbf{Z}(\pi_1))$. (b) π_1 and π_2 cannot dominate each other. For example, there exists $f(\mathbf{z}) = \mathrm{ReLu}(z_0 - 5.0)$, such that $\mathbb{E}f(\mathbf{Z}(\pi_2)) > \mathbb{E}f(\mathbf{Z}(\pi_1))$, thus π_1 cannot dominate π_2. Similarly, π_2 cannot dominate π_1. Thus there is no dominant relationship between the two policies. (c) π_1 and π_2 cannot dominate each other, which is similar to (b).

The definition of stochastic dominance extends univariate first-order stochastic dominance [9] into multivariate cases. Figure 2 provides some examples of stochastic dominance given 2D returns of two policies. The definition of stochastic dominance in policies allows for comparing the optimality of distributions between policies, which allows us to define the Distributionally Pareto-optimal policy:

Definition 3 (Distributionally Pareto-Optimal Policy)**.** *Formally, π_1 is Distributionally Pareto-Optimal Policy if there does not exist a policy π_2 such that $\boldsymbol{\mu}(\pi_2) \succ_{SD} \boldsymbol{\mu}(\pi_1)$.*

In other words, π_1 is considered Distributionally Pareto-Optimal (DPO) if it is not stochastically dominated by any other policy. This implies that π_1 offers the highest expected utility under some non-decreasing utility function f and cannot be outperformed by any other policy in the problem. DPO policies are essential in our framework as they represent the most desirable policies for users with different distributional preferences. To find such a policy, we have the following theorem:

Theorem 1. *If a policy π has optimal expected utility under some non-decreasing utility function f, then it is a Distributionally Pareto-Optimal policy. Also, any Pareto-Optimal policy is a Distributionally Pareto-Optimal policy.*

This theorem guarantees that a policy achieving the highest expected utility for a given utility function will be a DPO policy, making it a suitable candidate for deployment in MORL problems. In the next section, we base on the result of this theorem to find optimal policies for a diverse set of utility functions, in order to learn the set of DPO policies. This result also shows that our definition of Distributional Pareto-Optimal policies is an extension of Pareto-optimal policies, allowing for a more diverse set of optimal policies to be captured.

We also formally prove that optimal policies for risk-sensitive and safe constraint objectives belong to the set of Pareto-Optimal policies. This proves that DPO policies can successfully cover policies with diverse distributional preferences. The detailed proof is provided by Theorem 2 in Appendix A.

4.2 Distributionally Pareto-Optimal Multi-Objective Reinforcement Learning

We now present our proposed algorithm, Distributionally Pareto-Optimal Multi-Objective Reinforcement Learning (DPMORL). The main idea of DPMORL is to learn a set of non-linear utility functions that can guide the agent to discover distributional Pareto-optimal policies. The algorithm proceeds in two stages: (1) generating the utility functions and (2) training the policies.

4.2.1 Utility Function Generation with Diversity-based Objective

The first component of our algorithm focuses on generating a diverse set of plausible utility functions, upon which we find the optimal policies to find a diverse set of optimal policies. This is essential to ensure our method can accommodate various distributional preferences and adapt to different problem settings. To achieve this, we propose (1) Non-decreasing Neural Network for parameterizing a diverse set of non-linear utility functions (2) an objective function of minimum distance which encourages generating a diverse set of utility functions.

Non-decreasing Neural Network. We employ non-decreasing neural network to parameterize the utility function. The use of a neural network allows us to represent complex, non-linear, and arbitrary continuous utility functions, while the non-decreasing constraint ensures that the utility function satisfies the desired properties for multi-objective problems. We ensure the non-decreasing property in neural networks by constraining the weight matrix to be non-negative and the activation function to be non-decreasing following existing work in convex neural networks [28] and QMIX [29], which can approximate any multivariate non-decreasing function with arbitrary small errors [30]. The implementation details of the Non-decreasing Neural Network are provided in Appendix B.

Diversity-based Objective Function. We propose an objective function based on diversity for learning a diverse set of plausible utility functions. Specifically, we define $f_{\theta_1}, f_{\theta_2}, \cdots, f_{\theta_M}$ to be the set of candidate utility functions, parameterized by $\theta_1, \cdots, \theta_M$. For any given utility function f_{θ_i}, the learning objective is defined as

$$J^{\text{value}}(\theta_i) = \min_{j \neq i} \mathbb{E}_{\boldsymbol{z} \sim \mathcal{U}([0,1]^K)} [f_{\theta_i}(\boldsymbol{z}) - f_{\theta_j}(\boldsymbol{z})]^2 \tag{1}$$

$$J^{\text{grad}}(\theta_i) = \min_{j \neq i} \mathbb{E}_{\boldsymbol{z}_1, \boldsymbol{z}_2 \sim \mathcal{U}([0,1]^K)} \left[\frac{f_{\theta_i}(\boldsymbol{z}_2) - f_{\theta_i}(\boldsymbol{z}_1)}{\|\boldsymbol{z}_2 - \boldsymbol{z}_1\|} - \frac{f_{\theta_j}(\boldsymbol{z}_2) - f_{\theta_j}(\boldsymbol{z}_1)}{\|\boldsymbol{z}_2 - \boldsymbol{z}_1\|} \right]^2 \tag{2}$$

$$J(\theta_i) = \alpha J^{\text{value}}(\theta_i) + (1 - \alpha) J^{\text{grad}}(\theta_i) \tag{3}$$

Optimizing this objective function can encourage diversity among the generated utility functions within range $[0,1]^K$ in both value and derivative, leading to more comprehensive coverage of potential user preferences. We maximize the objective function by gradient descent on non-decreasing neural networks to generate a set of N utility functions.

4.2.2 Optimizing Policy with Utility-based Reinforcement Learning

Once we have generated a diverse set of utility functions, the second component of our algorithm focuses on optimizing policies to maximize the expected utility. This process, known as utility-based RL, leverages the generated utility functions to guide the optimization of policies. By focusing on the expected utility, our method can efficiently balance the trade-offs between multiple objectives and distributions, ultimately yielding policies that are more likely to align with user preferences.

We show that the following utility-based reinforcement learning algorithm can effectively optimize the policy with respect to a given utility function.

Algorithm 1 Utility-based Reinforcement Learning

Input: policy π, an environment $\mathcal{M} = (S, A, P, P_0, \boldsymbol{R}, \gamma, T)$, utility function f
Output: new policy π'

1: Augment state space with $\tilde{S} = \mathcal{S} \times \mathcal{Z}$, where $\mathcal{Z}$ is the space of cumulative multivariate returns.
2: Let transition $\tilde{P}_0(\cdot)$ and $\tilde{P}(\cdot|(s_t, \boldsymbol{z}_t), a_t)$ with $s_0 \sim P(s_0), z_0 = 0$, and $s_{t+1} \sim P(\cdot|s_t, a_t)$, $\boldsymbol{z}_{t+1} = \boldsymbol{z}_t + \gamma^t \boldsymbol{r}_t$.
3: Let scalar reward function R as $R((s_t, \boldsymbol{z}_t), a_t, (s_{t+1}, \boldsymbol{z}_{t+1})) = \gamma^{-t}[f(\boldsymbol{z}_{t+1}) - f(\boldsymbol{z}_t)]$.
4: Optimize policy π under environment $\tilde{\mathcal{M}} = (\tilde{S}, A, \tilde{P}, \tilde{P}_0, R, \gamma, T)$ under off-the-shelf RL algorithm (such as PPO or SAC) to π'.

Briefly, Algorithm 1 augments the state space with the cumulative multi-objective returns, and transforms the multi-dimensional rewards into a scalar reward by the difference in the utility function f. The following result shows that the new scalar-reward environment $\tilde{M}$ generated by Algorithm 1 has the same optimal policy as the optimal policy under utility function f:

Theorem 2. *The optimal policy π^* under environment $\tilde{\mathcal{M}} = (\tilde{S}, A, \tilde{P}, \tilde{P}_0, R, \gamma, T)$, with scalar reward function*

$$R((s_t, \boldsymbol{z}_t), a_t, (s_{t+1}, \boldsymbol{z}_{t+1})) = \gamma^{-t}[f(\boldsymbol{z}_{t+1}) - f(\boldsymbol{z}_t)]$$

is the optimal policy in the utility function f, i.e.

$$\mathbb{E}_{\boldsymbol{z} \sim \boldsymbol{\mu}(\pi^*)} [f(\boldsymbol{z})] = \max_{\pi} \mathbb{E}_{\boldsymbol{z} \sim \boldsymbol{\mu}(\pi)} [f(\boldsymbol{z})] .$$

An advantage of Algorithm 1 is that we can directly utilize off-the-shelf RL algorithms, such as PPO [31] and SAC [32] without any modification, which makes the algorithm easy to implement using widespread existing implementations of online RL algorithms.

It is also important to note that our algorithm simplifies to optimizing the weighted sum of rewards in MORL when the utility function is linear. This implies that our method is a generalization of the linear utility function MORL approaches by accommodating a wide range of non-linear utility functions. This flexibility makes our algorithm particularly suited for problems where the user's preferences may not be adequately captured by a linear utility function.

4.2.3 Iterative Generation of Utility Function and Policies Optimization

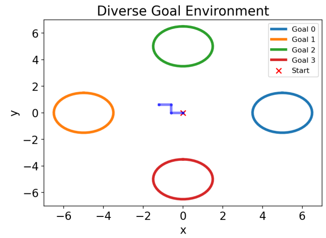
The range of multi-objective returns may not be known prior to policy optimization, which could make it challenging to generate relevant utility functions for a specific task. To mitigate this issue, we introduce an iterative process that alternates between generating currently plausible utility functions and optimizing policies based on these functions. During each iteration, we gather return samples throughout the optimization process and update the utility function. This update is particularly aimed at enhancing the diversity of the utility function with respect to the returns observed during the optimization process. A detailed outline of the algorithm can be found in Appendix C.

In the next experimental section, DPMORL only undergoes a single iteration: we initially generate a set of N utility functions as per the methodology detailed in Section 4.2.1. Subsequently, it optimizes a set of N policies using these generated utility functions, as outlined in Section 4.2.2. In Appendix C, we provide a case study demonstrating the application of DPMORL in an iterative training context.

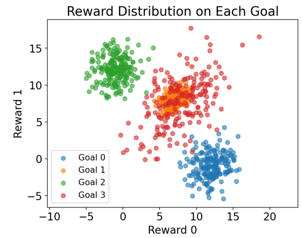
5 Experiments

In this section, we conducted several experiments under the setting of MORL. Through the experiments, we want to investigate the following questions:

1. Can Utility-based RL effectively learn policies with diverse distributional preferences?
2. Can DPMORL generate a set of diverse non-linear utility functions?
3. Can DPMORL obtain promising performance compared with state-of-the-art MORL methods in view of expected and distributional preferences?

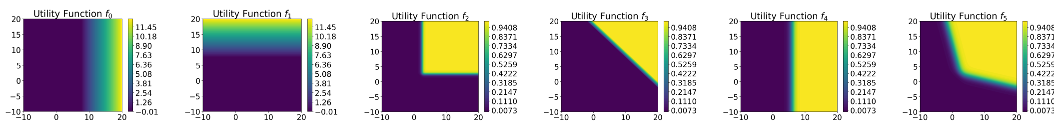


(a)

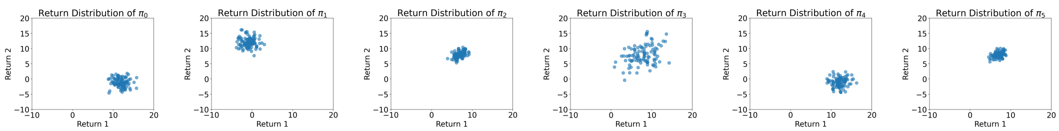


(b)

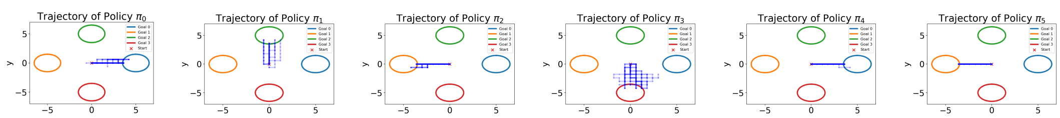
Figure 3: (a) The map of the DiverseGoal environment. (b) Reward distribution within each goal position.



(a)



(b)



(c)

Figure 4: (a) Utility function used for training policies. (b) Return distribution of policy learned by Algorithm 1 to maximize expected utility for each utility function. (c) The trajectory of policies learned by maximizing expected utility for each utility function.

5.1 Case Study on DiverseGoal Environment

To answer the first question, we train policies with a diverse set of utility functions on DiverseGoal, an environment that provides a diverse set of reward distributions.

DiverseGoal is a MORL environment in Figure 3 with multiple goals that the agent can take multiple steps to reach, where each goal has its unique reward distributions. Upon reaching a particular goal, the agent secures a 2D reward. This reward is sampled from the specific distribution associated with that goal, as illustrated in Figure 3b. Conversely, if the agent reaches the boundary of the map, it incurs a negative 2-dimensional reward. The environment requires the agent to navigate the trade-offs between various reward distributions, underscoring the complexity and nuance inherent to the task.

Here, we focus on showing the effectiveness of the Utility-based RL algorithm (Algorithm 1). We select six different non-linear functions $f_0, f_1, \cdots, f_5$, each in favor of distributions at one goal. We train policy π_i with utility function f_i with Algorithm 1 where $i = 0, 1, \cdots, 5$, and show the policy's trajectory and return samples. Ideally, the learned policy should reach the goal that yields the highest expected utility under the utility function. The results are illustrated in Figure 3. Under different type of utility function and return distributions, the utility-based RL algorithm is able to find the optimal policy with highest expected utility, which shows the effectiveness of the utility-based RL algorithm.

5.2 Main Experiment

In this section, we illustrate the performance of DPMORL on five environments based on MO-Gymnasium [33] to answer the latter two questions.

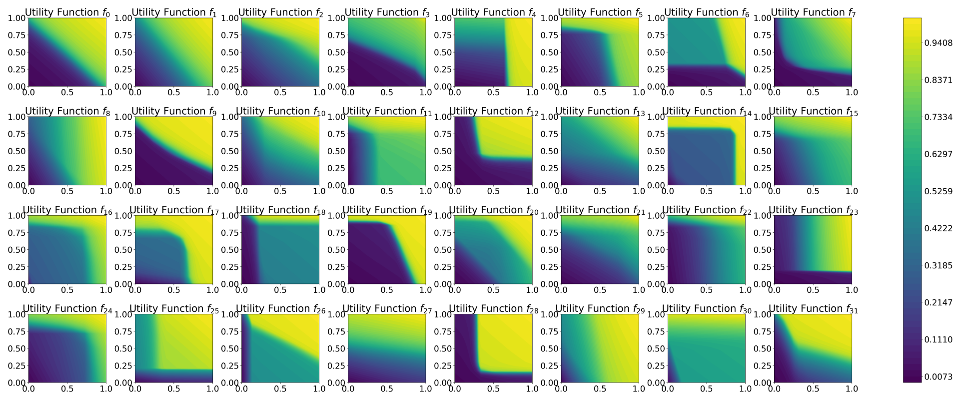


Figure 5: Illustration of 2D utility functions learned by our methods in Section 4.2.1.

Environments. We conducted experiments across five environments based on MO-Gymnasium [33] to evaluate the performance of our proposed method, DPMORL. These environments represent a diverse range of tasks, from simple toy problems to more complex continuous control tasks, and cover various aspects of multi-objective reinforcement learning:

- DeepSeaTreasure: A classic MORL benchmark that requires exploration in a gridworld to find treasures with different values and depths.
- FruitTree: A multi-objective variant of the classic gridworld problem, where an agent has to collect different types of fruits with varying rewards and penalties.
- HalCheetah, Hopper, MountainCar: Three continuous control tasks that require controlling different agents for task solving and minimizing energy usage.

More details about the environments are gathered in Appendix C.

Baselines. We compare DPMORL with four state-of-the-art baselines in the context of distributional preferences of MORL: Optimistic Linear Support (**OLS**) [34, 12]; Prediction-Guided Multi-Objective Reinforcement Learning (**PGMORL**) [4]; Generalized Policy Improvement with Linear Support (**GPI-LS**) [3]; Generalized Policy Improvement with Prioritized Dyna (**GPI-PD**) [3].

Training Details. For all methods, including DPMORL and the baselines, each policy was trained for 1×10^7 steps. We learn a set of $N = 20$ policies for DPMORL and all of the baselines to ensure a fair comparison. Finally, we use the learned $N = 20$ policies in each method for evaluations.

Implementation Details. We use PPO algorithms implemented in Stable Baselines 3 [35] in Algorithm 1. We use a normalization technique by linearly mapping the return into scale $[0, 1]^K$ without modifying the optimal policies. Detailed implementations are provided in Appendix B.

Evaluation Metrics. To thoroughly evaluate the performance of DPMORL and compare it with the baseline methods, we employed four distinct metrics. These comprise two existing MORL metrics, HyperVolume and Expected Utility, in addition to two novel metrics developed specifically for assessing distributional preference, i.e., Constraint Satisfaction and Variance Objective. The latter two are designed to underscore the optimality of multivariate distributions associated with the learned policies. In terms of Constraint Satisfaction, we randomly generate $M = 100$ constraints for the policy set produced. The Constraint Satisfaction metric is then computed by considering the highest probability within the policy set that satisfies each individual constraint. The Variance Objective metric, on the other hand, involves generating $M = 100$ random linear weights. These weights are applied to both the expected returns and the standard deviations of returns in each dimension. This objective encourages attaining greater expected returns while simultaneously reducing variance, thereby catering to dynamic preferences. Further details about the implementation of these evaluation metrics are provided in Appendix B.

5.3 Results

Since PGMORL works on continuous action spaces, here we omit the results of PGMORL on environments with discrete action space, DeepSeaTreasure and FruitTree. More results are gathered in Appendix C.

Table 1: Experimental results of each method on each standard MORL metric on all five environments. "EU" stands for Expected Utility, and "HV" stands for Hypervolume.

Environment	Hopper		HalfCheetah		MountainCar		DeepSeaTreasure		FruitTree	
Metric	EU	HV	EU	HV	EU	HV	EU	HV	EU	HV
GPI-PD	2374.44	5033458.34	412.62	1083227.57	-55.44	7367.59	5.93	9.75	**6.98**	0.14
GPI-LS	1398.00	1446705.10	98.31	2839051.60	-37.37	8052.58	5.04	9.36	3.84	1.49
OLS	175.07	3298.13	-580.38	1420270.13	-470.00	2.45	4.64	**10.28**	6.27	7.49
PGMORL	300.19	32621.25	111.30	383681.25	-429.48	94.45	-	-	-	-
DPMORL (Ours)	**3492.93**	**12154967.99**	**1189.68**	**8593769.62**	**-29.89**	**8663.80**	**6.70**	8.68	6.89	**16.39**

Table 2: Experimental results of each method on Distributional metric on all five environments. "Constraints" stands for Constraints Satisfaction, and "Var" stands for Variance Objective.

Environment	Hopper		HalfCheetah		MountainCar		DeepSeaTreasure		FruitTree	
Metric	Constraints	Var	Constraint	Var	Constraint	Var	Constraint	Var	Constraint	Var
GPI-PD	0.47	979.74	0.64	83.49	**1.00**	-31.15	0.85	**2.59**	0.65	**3.42**
GPI-LS	0.25	607.47	0.60	51.67	**1.00**	-22.73	0.85	1.99	0.33	1.35
OLS	0.05	75.48	0.43	-311.27	0.05	-244.21	0.80	1.86	0.50	2.98
PGMORL	0.05	98.47	0.50	45.48	0.11	-249.69	-	-	-	-
DPMORL (Ours)	**0.76**	**1645.89**	**0.82**	**431.26**	**1.00**	**-16.32**	**0.90**	2.54	**0.67**	3.21

Generation of Utility Functions. In accordance with the methodology detailed in Section 4.2.1, we employ non-decreasing neural networks in conjunction with diversity-focused objectives to generate a diverse assortment of utility functions. The generated functions are visually represented in Figure 5. The outcomes clearly indicate that optimizing diversity-based objective functions allows for generating a broad range of non-linear utility functions, thereby encompassing an expansive array of preferences with respect to returns. Subsequently, we utilize the first $N = 20$ utility functions depicted in Figure 5 to train an equivalent number of policies under DPMORL.

Standard MORL Metrics. The results under standard MORL metrics are shown in Table 1. Focusing on the Expected Utility, the performance of DPMORL is the highest in the Hopper, HalfCheetah, MountainCar, and DeepSeaTreasure environments (4/5), indicating that our method outperforms others in terms of expected utility. While for FruitTree environment, DPMORL also obtains consistent performance with the best contender, GPI-LS. In terms of the HyperVolume, DPMORL also performs the best in the Hopper, HalfCheetah, MountainCar, and FruitTree environments (4/5). The results show that DPMORL can yield better performance across both metrics and most environments (Hopper, HalfCheetah, MountainCar, and DeepSeaTreasure), meanwhile delivering robust and consistent results on the other one (FruitTree).

Distributional MORL Metrics. The results under distributional metrics are shown in Table 2. DPMORL outperforms other methods in most environments on both Constraint Satisfaction (5/5) and Variance Objective (3/5), indicating its strong ability to handle the distributional multi-objective reinforcement learning problem. For the rest environments, DPMORL also obtained comparable results compared with the best contender, GPI-LS. The results show that the policies learned by DPMORL have a higher probability of satisfying randomly generated constraints, and can better balance the trade-off between expectations and variances.

Return Distributions of Learned Policies by DPMORL. We provide visualizations of the multi-dimensional return distributions of each policy learned by DPMORL on all five environments. On Hopper, HalfCheetah, DeepSeaTreasure and FruitTree, DPMORL learns a diverse set of policies with different distributional properties in returns. We also visualize the learning process of different policies. We provide these results in Appendix C.

6 Conclusion

In this work, we initialized the study of distributional MORL, specifically when preferences over different objectives and their return distributions are uncertain. We introduced the concept of Distributional Pareto-Optimal (DPO) policies with rigorous theoretical analysis, which extend the notion of Pareto optimality in MORL to include preferences over the entire distribution of returns, not just their expected values. To obtain such desirable policies, we proposed a new algorithm, DPMORL, designed to learn DPO policies with non-linear utility functions. DPMORL allows for expressing a wide range of distributional preferences, providing a flexible and expressive approach to MORL. Experiment results showed that DPMORL consistently outperformed existing MORL methods in terms of optimizing policies for multivariate expected and distributional preferences.

References

[1] Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

[2] Conor F. Hayes, Roxana Radulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel de Oliveira Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective reinforcement learning and planning. *Auton. Agents Multi Agent Syst.*, 36(1):26, 2022.

[3] Lucas Nunes Alegre, Ana L. C. Bazzan, Diederik M. Roijers, Ann Nowé, and Bruno C. da Silva. Sample-efficient multi-objective learning via generalized policy improvement prioritization. *CoRR*, abs/2301.07784, 2023.

[4] Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. Prediction-guided multi-objective reinforcement learning for continuous robot control. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 10607–10616. PMLR, 2020.

[5] Anirban Majumdar and Claire J Tomlin. Risk-sensitive sequential decision making: Uncertainty-based performance metrics and risk-sensitive control. *Foundations and Trends® in Systems and Control*, 4(4):265–365, 2017.

[6] Shiau Hong Lim and Ilyas Malik. Distributional reinforcement learning for risk-sensitive policies. In *Advances in Neural Information Processing Systems*, 2022.

[7] Aivar Sootla, Alexander I Cowen-Rivers, Taher Jafferjee, Ziyan Wang, David H Mguni, Jun Wang, and Haitham Ammar. Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *International Conference on Machine Learning*, pages 20423–20443. PMLR, 2022.

[8] Mathieu Reymond, Conor F Hayes, Denis Steckelmacher, Diederik M Roijers, and Ann Nowé. Actor-critic multi-objective reinforcement learning for non-linear utility functions. *Autonomous Agents and Multi-Agent Systems*, 37(2):23, 2023.

[9] Haim Levy. Stochastic dominance and expected utility: Survey and analysis. *Management science*, 38(4):555–593, 1992.

[10] Moshe Shaked and J George Shanthikumar. *Stochastic orders*. Springer Science Business Media, 2007.

[11] Patrick Mannion, Jim Duggan, and Enda Howley. Dynamic treatment regimes in mobile health. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 526–533. IEEE, 2016.

[12] Lucas Nunes Alegre, Ana L. C. Bazzan, and Bruno C. da Silva. Optimistic linear support and successor features as a basis for optimal policy transfer. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 394–413. PMLR, 2022.

[13] Axel Abels, Diederik M. Roijers, Tom Lenaerts, Ann Nowé, and Denis Steckelmacher. Dynamic weights in multi-objective deep reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 11–20. PMLR, 2019.

[14] Matthieu Geist, Julien Pérolat, Mathieu Laurière, Romuald Elie, Sarah Perrin, Olivier Bachem, Rémi Munos, and Olivier Pietquin. Concave utility reinforcement learning: The mean-field game viewpoint. In Piotr Faliszewski, Viviana Mascardi, Catherine Pelachaud, and Matthew E. Taylor,

editors, *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*, pages 489–497. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022.

[15] Mridul Agarwal, Vaneet Aggarwal, and Tian Lan. Multi-objective reinforcement learning with non-linear scalarization. In Piotr Faliszewski, Viviana Mascardi, Catherine Pelachaud, and Matthew E. Taylor, editors, *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*, pages 9–17. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022.

[16] Mridul Agarwal, Vaneet Aggarwal, and Tian Lan. Multi-objective reinforcement learning with non-linear scalarization. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 9–17, 2022.

[17] Marc G Bellemare, Will Dabney, and R'emi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 449–458. JMLR. org, 2017.

[18] Will Dabney, Mark Rowland, Marc G Bellemare, and Remi Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[19] Gabriel Barth-Maron, Matthew W. Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva TB, Alistair Muldal, Nicolas Heess, and Timothy P. Lillicrap. Distributed distributional deterministic policy gradients. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[20] Vivek S Borkar. Risk-sensitive reinforcement learning. *SIAM Journal on Control and Optimization*, 40(3):818–834, 2002.

[21] Aviv Tamar, Yonathan Glassner, and Shie Mannor. Optimality and approximation in online and risk-averse markov decision processes. *The Journal of Machine Learning Research*, 16(1):2553–2592, 2015.

[22] L A Prashanth and Shalabh Bhatnagar. Risk-sensitive reinforcement learning. *arXiv preprint arXiv:1604.00147*, 2016.

[23] Javier Garc'ia and Fernando Fern'andez. Safe exploration in markov decision processes. *arXiv preprint arXiv:1205.1111*, 2012.

[24] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. *arXiv preprint arXiv:1705.10528*, 2017.

[25] Yinlam Chow, Ofir Nachum, Edgar A Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1805.08711*, 2018.

[26] Zsolt Gabor and Zoltan Koles. Multi-objective reinforcement learning: A comprehensive overview. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, 3:2030–2035, 1998.

[27] Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707*, 2016.

[28] Brandon Amos, Lei Xu, and J Zico Kolter. Input convex neural networks. In *International Conference on Machine Learning*, pages 146–155. PMLR, 2017.

[29] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1):7234–7284, 2020.

[30] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating functional knowledge in neural networks. *Journal of Machine Learning Research*, 10(6), 2009.

[31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[32] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR, 2018.

[33] Lucas N. Alegre, Florian Felten, El-Ghazali Talbi, Grégoire Danoy, Ann Nowé, Ana L. C. Bazzan, and Bruno C. da Silva. MO-Gym: A library of multi-objective reinforcement learning environments. In *Proceedings of the 34th Benelux Conference on Artificial Intelligence BNAIC/Benelearn 2022*, 2022.

[34] Diederik M Roijers. Multi-objective decision-theoretic planning. *AI Matters*, 2(4):11–12, 2016.

[35] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.