Robust Multi-Objective Reinforcement Learning with Dynamic Preferences

Francois Buet-Golfouse UCAHFBU@UCL.AC.UK
University College London London, United Kingdom

Parth Pahwa PARTH.PAHWA95@GMAIL.COM
Independent Researcher London, United Kingdom

Editors: Emtiyaz Khan and Mehmet Gonen

Abstract

This paper considers multi-objective reinforcement learning (MORL) when preferences over the multiple tasks are not perfectly known. Indeed, it is often the case in practice that an agent is trying to achieve tasks that may have competing goals but does not exactly know how to trade them off. The goal of MORL is thus to learn optimal policies under a set of possible preferences leading to different trade-offs on the Pareto frontier. Here, we propose a new method by considering the *dynamics* of preferences over tasks. While this is a more realistic setup in many scenarios, more importantly, it helps us devise a simple and straightforward approach by considering a surrogate state space made up of both states and preferences, which leads to a joint exploration of states and preferences. Static (and possibly unknown) preferences can also be understood as a limiting case of our framework. In sum, this allows us to devise both deep Q-learning and actor-critic methods based on *planning* under a preference-dependent policy and *learning* the multi-dimensional value function under said policy. Finally, the performance and effectiveness of our method are demonstrated in experiments run on different domains.

1. Introduction

In comparison to traditional Reinforcement Learning ("RL"), problems where rewards are scalar and a single best solution exists, real world problems are inherently multi-objective, insofar as an agent learns to optimise over multiple, often competing, tasks. For example, in an online marketing campaign an agent may try to maximise customer reach while trying to minimise the campaign expenditure. In short, the goal of an agent is to learn an optimal policy under a set of preferences expressing the relative importance of each objective.

To deal with multi-objective problems, researchers have tried to incorporate preferences as a fixed choice and scalarise rewards (Konak et al., 2006; Lin, 2005; Mossalam et al., 2016), thereby reducing the problem to a single objective optimisation task. While this approach is well researched, it only addresses the subset of problems where the preferences over the objectives are known beforehand. A tangential strategy is to learn a set of optimal policies that can span the space of preferences. These methods were addressed in Natarajan and Tadepalli (2005); Barrett and Narayanan (2008); Li et al. (2020), but generally lack scalability in high dimensional environments.

In this paper, we present a multi-objective Markov decision process ("MOMDP") framework incorporating a transition function over preferences, leading to an algorithm capable of learning a single parameterised policy encompassing the dynamics of preferences over tasks. This algorithm can be extended to existing state-of-the-art methods, such as deep Q-learning (Mnih et al., 2013) and actor-critic methods (Mnih et al., 2016), while overcoming the shortcomings of existing MORL methods. It is scalable as it optimises for a single parameterised preference-dependent policy but implements the principles of multi-task learning ("MTL") to also learn a multi-dimensional value function under the said policy. Furthermore, since our approach considers a surrogate state space made up of both states and preferences, we have devised a *joint* exploration strategy of states and preferences. Thus, we supplement the Q-learning algorithm with hindsight experience replay (Andrychowicz et al., 2017) for better sample efficiency and an exemplar network (Fu et al., 2017) for efficient exploration over the surrogate state space.

In Section 2, we introduce background concepts and our revised MOMDP with preference transition. In Section 3, we present our novel Robust Multi-Objective Reinforcement Learning with Dynamic Preferences ("RDP MORL") algorithm and provide theoretical guarantees. In Section 4, we show some empirical results and compare our algorithms performance with other state-of-the-art MORL algorithms.

2. Background And Related Work

2.1. Multi-objective Markov Decision Process

Let us first introduce the Markov framework for solving the multi-objective sequential decision problem. An MOMDP can be represented by the tuple $\langle \mathcal{S}, \mathcal{A}, P_S, \mathbf{r}, \gamma, \Omega, f_\omega, P_\omega \rangle$, where:

- $\mathcal{S}$ is the state space,
- $\mathcal{A}$ is the action space,
- $P_S : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function over the state space,
- $\mathbf{r} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^d$ is the vector reward function specifying rewards for $d \geq 1$ objectives,
- γ is the discount factor and $\gamma \in [0, 1)$,
- Ω is the space of preferences where $\omega \in \Omega$ s.t. $\sum_{i=1}^{d} \omega_i = 1$ and $\omega_i \geq 0$ for $d \geq 1$ objectives,
- $f_\omega : \mathbf{r} \to \mathbb{R}$ is the scalarisation function for preference $\omega \in \Omega$,
- $P_\omega : \Omega \times \Omega \to [0, 1]$ is the transition function over the preference space.

We note that if $d = 1$ and $P_\omega(\omega, \hat{\omega}) = 0$ if $\omega \neq \hat{\omega}$ and 1 if $\omega = \hat{\omega}$, the MOMDP collapses to a standard Markov decision process ("MDP")[1]. A policy is defined as a mapping $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ from state to action. The vector value of state s at time t under a policy π is given by the multidimensional value function, defined as

$$\mathbf{V}^\pi(s) = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^t \mathbf{r}_{t+i+1} | \pi, S_t = s\right], \tag{1}$$

1. In the case where we have $d > 1$ objectives and $P_\omega(\omega, \hat{\omega}) = 0$ if $\omega \neq \hat{\omega}$ and 1 if $\omega = \hat{\omega}$, we recover the classic formulation of the MOMDP.

where $\mathbf{r}_{t+1}$ is the reward received at time-step $t+1$. Similarly we can define the vectorised $\mathbf{Q}$ function as the expected long term reward by taking action a in state s at time t under a policy π:

$$\mathbf{Q}^{\pi}(s,a) = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^{t} \mathbf{r}_{t+i+1} | \pi, S_t = s, A_t = a\right]. \tag{2}$$

If we consider the set of all possible value functions, we can construct the Pareto frontier $\mathcal{F}^* := \{\mathbf{V}(s) | \nexists \mathbf{V}'(s) \geq \mathbf{V}(s)\}$ and under the space of all possible preferences in Ω, define the convergence set ("CCS") of the Pareto frontier as $\mathcal{CCS} := \{\mathbf{V}(s) \in \mathcal{F}^* | \exists \boldsymbol{\omega} \in \Omega \; s.t. \; \boldsymbol{\omega}^T \mathbf{V}(s) \geq \boldsymbol{\omega}^T \mathbf{V}'(s), \forall \; \mathbf{V}'(s) \in \mathcal{F}^*\}$, where $\boldsymbol{\omega} \in \mathbb{R}^d$ s.t. $\sum_{i=1}^{d} \omega_i = 1$ and $\omega_i \geq 0$.

Remark 1 *The convex convergence set is a subset of the Pareto frontier, i.e., $\mathcal{CCS} \subset \mathcal{F}^*$ and the Pareto front can be regarded as a set of non dominated policies, i.e., there exists no other policy that can improve the expected return for an objective without reducing the expected return of at least one different objective. Throughout this paper, we consider the scenario where the scalarisation function f_ω is linear, i.e. $f_\omega(\mathbf{r}) = \omega^T \mathbf{r}$.*

Problem Statement For any MOMDP there exists a set of policies corresponding to the convex convergence set ($\mathcal{CCS}$). Our goal is to train an agent to discover and act according to the preference dependent policy which spans the $\mathcal{CCS}$.

2.2. Related Work

Multi-Objective Reinforcement Learning MORL problems involve finding Pareto optimal solutions using a combination of multi-objective optimisation and reinforcement learning techniques. MORL algorithms follow either a single-policy or a multiple-policy (Vamplew et al., 2011) approach. Single-policy approaches seek to find the optimal policy by fixed preference induced scalarisation of the multi-objective problem. Researchers have explored the effects of both linear and non linear scalarisation (Van Moffaert et al., 2013). However, in reality the set of preferences may be unknown at training time or may change over time. Multiple-policy approaches focus on approximating the set of policies that span the Pareto frontier. The methodologies range from repeatedly calling single-policy MORL over different preferences (Natarajan and Tadepalli, 2005; Mossalam et al., 2016; Zuluaga et al., 2016), generalising the Q-learning update rule to multi-objective settings (Reymond and Nowé, 2019; Yang et al., 2019) or by modifying gradient based policy search (Parisi et al., 2014; Pirotta et al., 2015a,b). Recent works have demonstrated the application of MORL with deep reinforcement learning (Van Seijen et al., 2017; Friedman and Fontaine, 2018), in high dimensional state space (Abdolmaleki et al., 2020). In our approach, we evaluate on both discrete and continuous observation spaces.

Meta Learning and Multi-Task Learning These methods were explored by (Chen et al., 2019; Teh et al., 2017; Riedmiller et al., 2018; Wulfmeier et al., 2019) as a way of solving multi-objective control problems. Meta learning paradigm involves learning a general policy that is not Pareto optimal but computationally efficient and adapting to objective preferences while multi-task learning frameworks solve the MORL problems by jointly learning a separate policy. While our methodology does implement multi-task learning, we create learning tasks over the vectorised and scalar value functions and not the competing objectives.

Q-Learning The Q-Learning framework has also been extended to MORL framework (Mossalam et al., 2016; Yang et al., 2019). In particular, Scalarised Q-learning (Mossalam et al., 2016) uses a vector value function with scalar updates and searches over preferences. The scalar updates, which involves computing the inner product of the value with the preferences, lead to sample inefficiency and sub optimal MORL policies. While Envelope Q-learning (Yang et al., 2019) tries to address these shortcomings, our approach introduces the transition function over preferences and learning over the extended state space, leading to robust and faster learning. The key contributions that distinguishes our work, robust Q-learning with dynamic preferences (RDP Q-learning), from Yang et al. (2019) is the introduction of transition function over preferences, which allows the agent to adapt to dynamic preferences while it is performing actions in the environment, and the formulation of estimating both the vector and scalar value functions as a multi-task learning problem with shared parameters. Additionally, the introduction of surrogate state spaces allows efficient exploration of the preference space (generally leading to a faster computation of the value function).

Exploration and Exemplar Networks Exploration plays a fundamental role in RL systems and in the original deep Q-learning paper Mnih et al. (2013), the authors use ε-greedy exploration to overcome the challenge of sparse reward signals. However, ε-greedy or other undirected exploration methods can be exponential in the depth of the state space (Thrun, 1992). Given that our methodology increases the dimensionality by introducing a surrogate state space, undirected methods do not scale and lead to suboptimal performance. Additionally, the fundamental idea of multi-objective reinforcement learning is to understand the effect of preferences on agent policies. Therefore, an efficient exploration over the *surrogate* space allows us to explore over the preference space and is sample efficient. Hence, in our implementation, we supplement Q-learning with exemplar networks from Fu et al. (2017) for a sophisticated exploration of the preference space and provide an ablation study in Appendix A to showcase the performance boost.

3. Multi-objective reinforcement learning with dynamic preferences

We propose a new framework for multi-objective reinforcement learning called Robust Multi-Objective Reinforcement Learning with Dynamic Preferences ("RDP MORL"). Our key idea is to consider the dynamics of preferences over tasks, and learn over a surrogate state space, which is defined as a combination of both states and preferences. The aim is two-fold: (1) dynamics over preferences introduces robustness by taking into account disturbances in preferences, (2) creating the surrogate state space allows the agent to efficiently explore and approximate the Pareto frontier. While this framework can be applied to both deep Q-learning (Mnih et al., 2013) and actor critic methods (Mnih et al., 2016), we focus on the former and introduce robust Q-learning with dynamic preferences.

Robust Q-learning with dynamic preferences Our algorithm uses the vectorised Q-value function to allow both the (vector and scalar) Q-networks to simultaneously learn a set of policies. While Yang et al. (2019) uses a similar methodology, the convex envelope update and optimality filter they define render their approach high-dimensional and computationally complex. Our proposed RDP Q-learning overcomes this hurdle by taking

actions according to the scalarised Q-network during both learning and evaluation. This methodology results in outperforming competitor algorithms with fewer iterations.

3.1. Revisiting MORL

In addition to the previous discussion, two more considerations led us to our proposed algorithm. Firstly, preferences can shift over time Guiso et al. (2018). Secondly, current MORL algorithms do not exploit *directly* the fact that similar preferences should lead to similar Q-values (except at points of discontinuity).

3.1.1. Building a Surrogate State Space

The key insight is to consider a surrogate (or augmented) state space combining the state space $\mathcal{S}$ as well as the preference space Ω

$$\langle \mathcal{S}, \mathcal{A}, P_S, \mathbf{r}, \gamma, \Omega, f_\omega, P_\omega \rangle = \langle (\mathcal{S} \times \Omega), \mathcal{A}, P_S \otimes P_\omega, f_\omega(\mathbf{r}), \gamma \rangle \tag{3}$$

In particular, under our linear preference assumption, the reward function in the surrogate space can be simply defined as $r((s,\omega),a) = \mathbf{r}(s,a)^T\omega$. This leads us directly to a straightforward definition of the action-value function Q as:

$$Q^\pi((s,\omega),a) = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^t \mathbf{r}_{t+i+1}^T \omega_{t+i+1} \middle| \pi, S_t = s, \omega_t = \omega, A_t = a\right]. \tag{4}$$

Remark 2 *Importantly, in general,* $Q^\pi((s,\omega),a) \neq \mathbf{Q}^\pi((s,\omega),a)^T\omega$ *as preferences are themselves dynamic. In the particular case where* $P(\omega_{j+1} = \omega' | \omega_j = \omega) = \delta(\omega,\omega')$*, the equality is recovered.*

3.1.2. Implicit Bellman Operator

Having defined the surrogate state space $(\mathcal{S} \times \Omega)$ and the Q-value function, one can simply apply Q-learning to the surrogate MDP. In other words, it follows directly from standard Q-learning Sutton and Barto (2018) that the multi-objective optimality operator $\mathcal{T}$ can be defined as:

$$\left(\mathcal{T}Q\right)((s,\omega),a) := \mathbf{r}(s,a)^T\omega + \gamma \mathbb{E}_{(s',\omega') \sim P_S \otimes P_\omega(\cdot|s,\omega,a)} \left[\max_{a' \in \mathcal{A}} Q((s',\omega'),a')\right] \tag{5}$$

$$\left(\mathcal{T}Q\right)(\hat{s},a) = \mathbf{r}(\hat{s},a) + \gamma \mathbb{E}_{\hat{s}' \sim P_S \otimes P_\omega(\cdot|\hat{s},a)} \left[\max_{a' \in \mathcal{A}} Q(\hat{s},a')\right]. \tag{6}$$

This enables us to use standard Q-learning in a straightforward fashion, as the Q-value update at iteration $j+1$ can be written as:

$$Q_{j+1}(\hat{s}_j,a) \leftarrow (1-\alpha_j(\hat{s}_j,a))Q_j(\hat{s}_j,a) + \alpha_j(\hat{s}_j,a)\left[\mathbf{r_j}^T\omega_j + \gamma \max_{a'} Q_j(\hat{s}_{j+1},a')\right], \tag{7}$$

where $\alpha_j \in [0,1)$ is the chosen step size function.

Remark 3 *Note that, while we are focusing here on Q-learning, similar approaches to SARSA or policy gradients (and actor-critic methods) are possible.*

One of the conditions for a Q-learning algorithm to converge Melo (2001) is that $\sum_j \alpha_j(\hat{s}, a) = +\infty$ and $\sum_j \alpha_j(\hat{s}, a)^2 < +\infty$, for all $\hat{s}, a$. In other words, all state-preference-action triplets must be visited infinitely often. This thus represents a challenge in terms of exploring effectively the space of preferences.

3.1.3. Multi-Objective Q-Value Function

The optimal policy derived from Q-learning is $\pi^*(a|\hat{s}) = \mathbf{1}_{\{\arg\max_{a \in A} Q(\hat{s}, a)\}}$. To obtain the multivariate Q-value function, all we have to do is apply policy evaluation under π^*:

$$\mathbf{Q}_{j+1}(\hat{s}_j, a) \leftarrow (1 - \alpha_j(\hat{s}_j, a))\mathbf{Q}_j(\hat{s}_j, a) + \alpha_j(\hat{s}_j, a)\left[\mathbf{r_j} + \gamma \mathbf{Q}_j(\hat{s}_{j+1}, \arg\max_{a' \in A} Q_j(\hat{s}, a'))\right]. \quad (8)$$

Importantly, these updates can be carried out in parallel with Q-learning on the surrogate state space.

3.2. Learning Algorithm

We implement two Q-networks, namely a multi-objective Q-network and a policy Q-network. The aim of the multi-objective Q-network is to approximate the vectorised $\mathbf{Q}$-function to allow the network to learn simultaneously over multiple preferences while the agent acts according to the policy selection Q-network. Since both networks observe the dynamics of the same environment, we model the learning stage of both as tasks that are separate but still related. Multi-task learning is known to outperform single-task algorithms (Zhang and Yang, 2021) in these scenarios. Using the multi-task approach, we provide the algorithm for RDP Q-learning in Algorithm 1.

3.2.1. Multi-Task Q-learning

Let $\mathcal{L}^{\alpha}(\theta)$ be the loss associated with the Multi-Objective Q-network parameterised by θ and let $\mathcal{L}^{\beta}(\hat{\theta})$ be the loss associated with the policy Q-network parameterised by $\hat{\theta}$ where $\theta, \hat{\theta} \in \Theta$ ($\hat{\theta}$ is a sub-vector of θ). We define the multi-objective Q-network loss as:

$$\mathcal{L}^{\alpha}(\theta) = \mathbb{E}_{\hat{s},a}\left[\|\mathbf{y} - \mathbf{Q}(\hat{s}, a; \theta)\|_2^2\right] \quad (9)$$

where $\hat{s} = (s, \omega)$ is the surrogate state, $\mathbf{y} = \mathbb{E}_{\hat{s}'}[\mathbf{r} + \gamma \mathbf{Q}(\hat{s}', \hat{a}; \theta)], \hat{a} = \max_a Q(\hat{s}_{j+1}, a; \hat{\theta})$, which is estimated by sampling transitions from the replay buffer. The policy Q-network loss is defined as:

$$\mathcal{L}^{\beta}(\hat{\theta}) = \mathbb{E}_{\hat{s},a}\left[(y - Q(\hat{s}, a; \hat{\theta}))^2\right] \quad (10)$$

where $\hat{s} = (s, \omega)$ is the surrogate state and $y = \mathbb{E}_{\hat{s}'}[\omega^T \mathbf{r} + \gamma \max_{\hat{a}} Q(\hat{s}', \hat{a}; \hat{\theta})]$. This leads to a multi-task loss function for the overall network. We provide a framework for solving this in the following sections.

Algorithm 1 RDP Q-learning

Require: network $Q_{\hat{\theta}}$ and $\mathbf{Q}_{\theta}$, sampling distribution G_{ω}, transition distribution H_{ω}, replay buffer D.

for *episode* $= 1, ...N$ **do**
  Sample a linear preference $\boldsymbol{\omega_0} \sim G_{\omega}$
  for $t = 0, \cdots, M-1$ **do**
    Observe state s_t and construct surrogate state space $\hat{s}_t = (s_t, \boldsymbol{\omega_t})$
    $a_t = \begin{cases} \text{random action}, & \text{w.p. } \epsilon \\ \arg\max_a Q(\hat{s}_t, a; \hat{\theta}), & \text{w.p. 1-}\epsilon \end{cases}$
    Execute action a_t and observe reward $\mathbf{r_t}$, state s_{t+1} and sample $\omega_{t+1} \sim H_{\omega_t}$ [a]
    Store transition $(s_t, \omega_t, \mathbf{r_t}, s_{t+1}, \omega_{t+1})$
    Sample random minibatch of transitions of size N_{τ} $(s_t, \omega_t, \mathbf{r_t}, s_{t+1}, \omega_{t+1})$
    for $i = 1, \cdots, N_{\omega}$ **do**
      $\omega_i \sim G_{\omega}$
      $\omega_{i+1} \sim H_{\omega_i}$
      for $j = 1, \cdots, N_{\tau}$ **do**
        $\hat{s}_{i,j+1} = (s_{j+1}, \omega_{i+1})$
        $y_{ij} = \begin{cases} \mathbf{r}_j^T \omega_i, & \text{for terminal } s_{j+1} \\ \mathbf{r}_j^T \omega_i + \gamma \max_{\hat{a}} Q(\hat{s}_{j+1}, \hat{a}; \hat{\theta}), & \text{otherwise} \end{cases}$
        $\hat{\mathbf{y}}_{ij} = \begin{cases} \mathbf{r}_j, & \text{for terminal } s_{j+1} \\ \mathbf{r}_j + \gamma \mathbf{Q}(\hat{s}_{j+1}, \hat{a}; \theta), \ \hat{a} = \max_a Q(\hat{s}_{j+1}, a; \hat{\theta}) & \text{otherwise} \end{cases}$
      end for
    end for
    Update $Q_{\hat{\theta}}$ and $\mathbf{Q}_{\theta}$ by performing gradient descent according to equation 12.
  end for
end for

a. We simulate H_{ω_t} using a Dirichlet distribution with $\alpha = \omega$

3.2.2. An Adversarial Setup

To motivate our approach, we consider an adversarial approach to multi-task learning. Let K be the number of tasks, z_k represent the loss associated with task $k = 1, \cdots, K$ and w_k the corresponding weight.

Robustness to Uncertainty Given a vector of average individual task losses $z \in \mathbb{R}^K$ and a reference distribution over tasks $w \in \mathbb{S}^{K-1}$, the adversary maximises the overall loss but is constrained by the distance (chosen to be the Kullback-Leibler divergence here) to the reference distribution w. The adversary's problem can thus be written as

$$\max_{\delta \in \mathbb{S}^{K-1}} \delta \cdot z - \frac{1}{\eta} D_{\mathrm{KL}}(\delta || w), \tag{11}$$

where $\eta > 0$ is fixed. It is immediate to check that the solution vector is given by $\delta_k^* = \frac{w_k e^{\eta z_k}}{\sum_{k=1}^K w_k e^{\eta z_k}}$ for $k = 1, \cdots K$. Thus, the adversary's problem is recast as a robust optimisation problem and expressed in terms of distribution uncertainty Glasserman and Xu (2014).

Model Fitting Considering $\mathcal{L}^{\alpha}(\theta)$, the loss associated with the multi-objective Q-network parameterised by θ, and $\mathcal{L}^{\beta}(\hat{\theta})$ the loss associated with the policy Q-network parameterised by $\hat{\theta}$, where $\theta, \hat{\theta} \in \Theta$ ($\hat{\theta}$ is a sub-vector of θ), we frame our learning problem in an adversarial setting, leading to the following formulation, where $\eta > 0$, $w_k \geq 0$ for all k and $\sum_{k=1}^K w_k = 1$:

$$\min_{\theta, \hat{\theta}} \max_{\delta} \left(\delta_1 \mathcal{L}^{\alpha}(\theta) + \delta_2 \mathcal{L}^{\beta}(\hat{\theta}) - \frac{1}{\eta} \left[\delta_1 \log\left(\frac{\delta_1}{w_1}\right) + \delta_2 \log\left(\frac{\delta_2}{w_2}\right) \right] \right)$$
$$= \min_{\theta, \hat{\theta}} \frac{1}{\eta} \log\left(w_1 e^{\eta \mathcal{L}^{\alpha}(\theta)} + w_2 e^{\eta \mathcal{L}^{\beta}(\hat{\theta})} \right). \tag{12}$$

3.3. Model Architecture

Unlike Mossalam et al. (2016) and Yang et al. (2019), our algorithm uses two different networks. Since both networks interact with the same environment and are interdependent, we introduce a shared network structure. The shared network creates embeddings that benefit from the joint learning experience. It consists of 4 fully connected hidden layers with $20 \times (dim(\mathcal{S}) + d)$ hidden units each, where $\mathcal{S}$ is the state vector and d is the number of objectives. The input to the shared network is the surrogate state (i.e., the concatenation of the state and preference vector). The multi-objective Q-network stacks one output layer of size $d \times |\mathcal{A}|$ on top of the shared network where $|\mathcal{A}|$ is the cardinality of the action space. The policy Q-network takes as input the output of the multi-objective Q-network combined with the preference vector and contains 4 fully connected hidden layers with $20 \times d \times (|\mathcal{A}| \times +1)$ and the output is of size $|\mathcal{A}|$, see Figure 1.

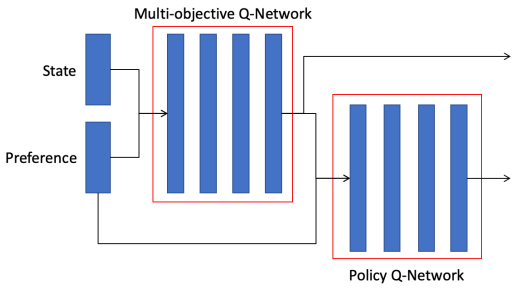

Figure 1: Network Architecture

4. Experiments

In this Section, we evaluate the performance of RDP Q-learning on three multi-objective reinforcement learning problems. We show how the algorithm can recover the optimal solution in the CCS, before comparing its performance against relevant benchmarks.

Evaluation Metric We use two metrics to evaluate the empirical performance on the problem domains:

- **Coverage Ratio** ("CR") evaluates the agents ability to recover the solutions from the finite convex convergence set. Let m be the number of objectives and $\mathcal{S} \subseteq \mathbb{R}^m$ the set of vector value functions recovered by the algorithm. We define $\mathcal{S} \cap_\epsilon CCS = \{\mathbf{V}^\pi \in \mathcal{S} | \exists \mathbf{V}^{\pi^*} \in CCS \text{ s.t. } ||\mathbf{V}^\pi - \mathbf{V}^{\pi^*}||_1 / ||\mathbf{V}^{\pi^*}||_1 \leq \epsilon\}$ Then the Coverage Ratio is calculated as the F-score, where $Precision = |\mathcal{S} \cap_\epsilon CCS|/|\mathcal{S}|$ and $Recall = |\mathcal{S} \cap_\epsilon CCS|/|CCS|$.

$$CR(S) = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{13}$$

- **Expected Utility Metric** ("EUM") evaluates the agent's ability to maximise user utility. It is defined as the expected maximum utility under the solution set S approximated by the algorithm:

$$EUM = \mathbb{E}[\max_{\pi_\omega \in S} f_\omega(\mathbf{V}^{\pi_\omega})] \tag{14}$$

Baselines: We compare RDP Q-learning's performance against its peer MORL algorithms: (1) *Envelope Q-learning* (Yang et al., 2019), a state of the art MORL algorithm that uses envelope Q updates to simultaneously learn multiple policies. It modifies deep

Q-Network for vector $\mathbf{Q}$ values. (2) *Scalarised Q-learning* (Mossalam et al., 2016), which uses scalarised Q-updates. (3) *MOFQI* (Castelletti et al., 2012), i.e., a multi-objective fitted Q-iteration with a large linear model as Q-approximator. (4) *CN+OLS* (Abels et al., 2019), which is a conditional neural network using an optimistic linear support method.

4.1. Fruit tree navigation

The Fruit tree navigation ("FTN") environment is a full binary tree of depth d ($d = 5, 6$ or 7), with a randomly assigned vector reward $\mathbf{r} \in \mathbb{R}^6$ on the leaf nodes, which are the terminal state. The reward vector encodes the values of six nutrition components {Protein, Carbs, Fats, Vitamins, Minerals, Water} in the leaf nodes. The rewards are designed to be Pareto optimal such that, for every leaf node, ω for which its reward is optimal, therefore all leaves lie on the CCS. The objective associated with the environment is to find a path from the root to a leaf node that maximises our overall utility for a given preference. At any non-terminating state in the tree, the agent has 2 actions available, choosing between the *left* or *right* subtree.

Table 1: Fruit Tree Coverage Ratio ($depth = 5$)

N_ω	Scalarised Q-learning	Envelope Q-learning	RDP Q-learning (2500 episodes)
1	0.9363 ±0.023	0.9706 ±0.027	**0.9980 ± 0.005**
4	0.9840 ±0.016	**1.0000 ± 0.000**	**1.0000 ± 0.000**
8	0.9968 ±0.007	**1.0000 ± 0.000**	**1.0000 ± 0.000**
16	**1.0000 ± 0.000**	**1.0000 ± 0.000**	**1.0000 ± 0.000**

Table 2: Fruit Tree Coverage Ratio ($depth = 6$)

N_ω	Scalarised Q-learning	Envelope Q-learning	RDP Q-learning (3000 episodes)
1	0.6250 ±0.057	0.9240 ±0.051	**0.9500 ± 0.012**
4	0.7654 ±0.077	0.9856 ±0.004	**0.9908 ± 0.008**
8	0.8560 ±0.067	0.9808 ±0.007	**0.9912 ± 0.009**
16	0.8976 ±0.062	0.9952 ±0.021	**0.9984 ± 0.003**

Table 3: Fruit Tree Coverage Ratio ($depth = 7$)

N_ω	Scalarised Q-learning	Envelope Q-learning	RDP Q-learning (3000 episodes)
1	0.5847 ±0.061	0.6000 ±0.029	**0.8728 ± 0.021**
4	0.6969 ±0.057	0.6544 ±0.066	**0.8034 ± 0.020**
8	0.6837 ±0.097	0.7437 ±0.040	**0.8326 ± 0.020**
16	0.6532 ±0.029	0.7936 ±0.015	**0.8243 ± 0.017**

We compare our model's performance against Envelope Q-learning and Scalarised Q-learning. We train all the three algorithms on FTN environment ($depth = 5, 6$ and 7) for 5000 episodes and N_ω sampled preferences during learning process. We compute the coverage ratio by testing the performance over 2000 episodes with randomly sampled preferences. The mean results over 5 trials for different depth are provided in Tables 1, 2, 3. We can see that RDP Q-learning is not only sample efficient but also able to outperform the baselines,

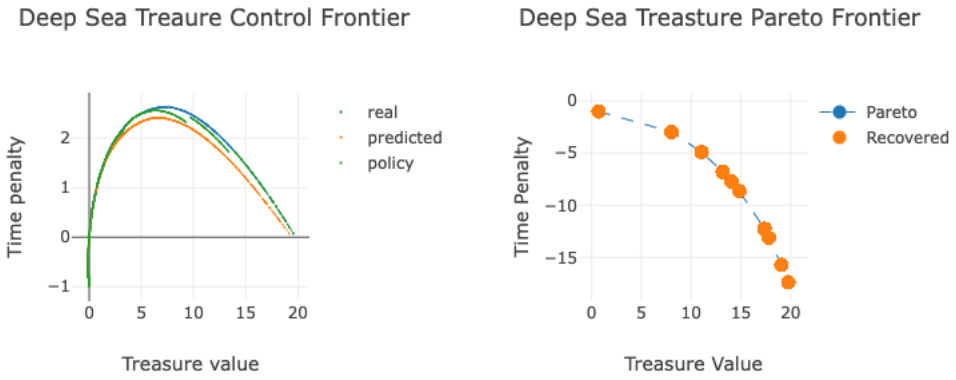

Figure 2: Illustration of the true and the RDP Q-learning recovered CCS for the deep sea treasure environment.

even when trained on significantly less episodes: 2500 on depth 5, 3000 on depth 6 and 3000 on depth 7.

4.2. Deep sea treasure

Deep sea treasure ("DST"), a classic MORL benchmark, is an episodic problem which was created to highlight the limitations of scalarisation (Vamplew et al., 2011). The environment is a 10×11 treasure hunt grid, with the agent controlling a submarine. There are multiple treasure locations with variable treasure values and two associated objectives, (1) Minimise the time taken to reach the treasure and (2) maximise the value of the treasure. We use the treasure values provide in Yang et al. (2019) to ensure the Pareto frontier is convex. For each episode, the agent is placed on the top left corner of the grid and has 4 actions available: *Up, Down, Left, Right.* The reward received by the agent is a 2 element vector, where the first element is the *time penalty* (computed by adding -1 on all turns), and the second element is the *treasure value.* The episode terminates when agent reaches a treasure state.

Table 4: Deep Sea Treasure

Method	Reference	Coverage Ratio
Envelope Q-learning	Yang et al. (2019)	0.994 ±0.001
Scalarised Q-learning	Mossalam et al. (2016)	0.989 ±0.024
CN+OLS	Abels et al. (2019)	0.751 ±0.163
MOFQI	Castelletti et al. (2012)	0.639 ±0.421
RDP Q-learning		**1.000 ± 0.000**

All agents are trained for 2000 episodes and we evaluate each algorithm for 2000 episodes with randomly sampled preferences. The mean coverage ratio results over 5 trials are provided in Table 4. The RDP Q-learning the best coverage ratio and is able to achieve after training for 1850 episodes. Figure 2 visualizes the optimal convex convergence set and RDP Q-learning approximated CCS. We can see that the RDP Q-learning solutions covers the entire optimal CCS.

4.3. Mountain Car

To our knowledge, the multi-objective version of the classic mountain-car task (Sutton, 1995) was first introduced in Vamplew et al. (2011). In the classical setting, the aim of the agent is to escape the car from the valley in minimum number of steps. The agent can perform 3 different actions: (1) Not accelerate, (2) Accelerate to the right and (3) Reverse to the left. Since the car's engine is less powerful than gravity, the agent must reverse to the left to build enough potential energy to escape from the right end. In the single objective setting, the agent receives a reward of -1 for all non terminating states.

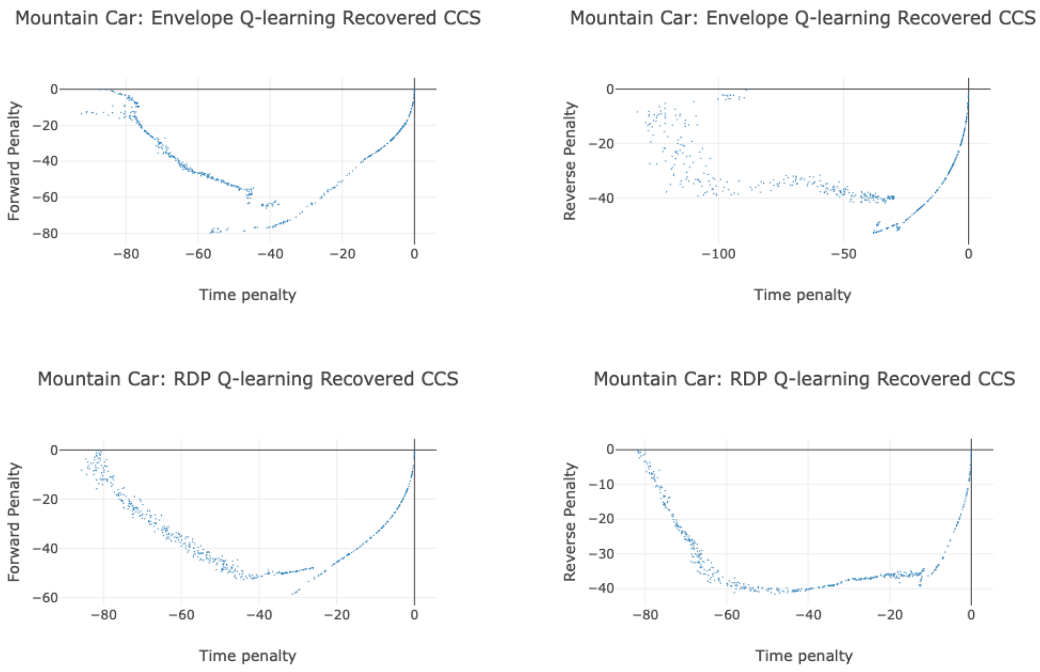

Figure 3: Illustration of the Envelope Q-learning and RDP Q-learning recovered CCS for the Mountain Car environment created by randomly sampling 500 preferences and calculating mean value over 50 trials

Vamplew et al. (2011) extends the objective space by introducing two additional objectives: minimise the number of (1) forward and (2) reverse accelerations and introduce a 3 dimensional vector reward where a penalty of -1 is received whenever one of the acceleration actions is executed. To increase the complexity of the optimisation problem we introduce an updated reward structure to provide a positive reinforcement when car displaces in the direction of the action performed by the agent. The reward structure is defined in Table 5. Unlike the FTN and DST environments, the state space of Mountain Car is continuous and has an unknown Pareto frontier, hence we empirically evaluate the performance using Expected Utility Metric.

Table 5: Mountain Car: Reward Structure

Time Penalty	Reverse Penalty	Acceleration Penalty	Action	Displacement
-1	0	0	No acceleration	None
-1	0	0	No acceleration	Left
-1	0	0	No acceleration	Right
-1	0	-1	Left	None
-0.5	0.5	-0.5	Left	Left
-1	0	-1	No acceleration	Right
-1	-1	0	Right	None
-0.5	-0.5	0.5	Right	Right
-1	-1	0	Right	Left

Table 6: Mountain Car: Expected Utility for different preferences

RDP Q-learning		Envelope Q-learning		Preference		
EUM	Steps	EUM	Steps	Time	Forward	Reverse
-77.42	105.72	-81.18	114.30	0.9	0.05	0.05
-28.58	300.00	-31.02	300.00	0.1	0.8	0.1
-27.53	300.00	-30.64	300.00	0.1	0.1	0.8
-27.27	169.79	**-11.57**	235.62	0.0	0.5	0.5
-66.48	125.88	-73.72	132.70	0.5	0.5	0.0
-56.91	169.79	-57.57	148.39	0.5	0.0	0.5

We train both algorithms for 1500 episodes and evaluate their performance on different sets of preferences. The mean results over 100 trials are provided in Tables 5-6. We can see –that across different preferences– RDP Q-learning has a higher expected utility value. The action behavior is fairly consistent for RDP Q-learning, whereas Envelope Q-network deviates from the optimal behaviour for the preference vector $[0.5, 0.5, 0.0]$ (Time, forward acceleration, reverse), by performing a higher count of forward acceleration (Right action). We also present the recovered CCS across the 3 objective pairs for both the RDP Q-learning and the baseline in Figure 3. The recovered CCS for the baseline is neither smooth nor convex, whereas RDP Q-learning retrieves a more consistent CCS.

5. Conclusion

In this paper, we have proposed a multi-objective reinforcement learning (MORL) framework that considers a surrogate (or augmented) state space made up of both states and

Table 7: Mountain Car: Count of actions performed for different preferences

RDP Q-learning			Envelope Q-learning			Preference		
Left	Right	None	Left	Right	None	Time	Forward	Reverse
40.41	65.31	0.00	37.16	77.02	0.12	0.9	0.05	0.05
254.43	0.00	47.57	298.07	0.00	3.93	0.1	0.8	0.1
2.41	201.84	97.75	0.00	296.15	5.85	0.1	0.1	0.8
32.03	45.29	92.47	14.33	17.82	203.47	0.0	0.5	0.5
66.65	59.11	0.12	53.10	71.89	7.71	0.5	0.5	0.0
20.63	112.80	3.50	20.41	125.80	2.18	0.5	0.0	0.5

preferences over the objectives. By designing an implicit Markov Decision Process based on this surrogate state space, we allow for *dynamic* preferences and have explored learning and planning via Q-learning under this particular formulation. In addition, as noticed in previous research, exploring the space of preferences is key to deriving optimal policies (and approximating the Pareto frontier), which we achieve here by encouraging the *joint* exploration of states and preferences. This is is further facilitated by adding exemplar rewards. This approach turns out to be particularly sample-efficient and robust. Finally, we demonstrated the effectiveness of the proposed framework by achieving improved performance against other state-of-the-art MORL algorithms on three different environments. Further research points to extending our approach to other algorithms such as actor-critic method, as well as exploring the impact of various types of preference dynamics on policy choices.

References

Abbas Abdolmaleki, Sandy Huang, Leonard Hasenclever, Michael Neunert, Francis Song, Martina Zambelli, Murilo Martins, Nicolas Heess, Raia Hadsell, and Martin Riedmiller. A distributional view on multi-objective policy optimization. In *International Conference on Machine Learning*, pages 11–22. PMLR, 2020.

Axel Abels, Diederik Roijers, Tom Lenaerts, Ann Nowé, and Denis Steckelmacher. Dynamic weights in multi-objective deep reinforcement learning. In *International Conference on Machine Learning*, pages 11–20. PMLR, 2019.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Leon Barrett and Srini Narayanan. Learning all optimal policies with multiple criteria. In *Proceedings of the 25th international conference on Machine learning*, pages 41–47, 2008.

Francois Buet-Golfouse and Islam Utyagulov. Towards fair unsupervised learning. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 1399–1409, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393522.

Andrea Castelletti, Francesca Pianosi, and Marcello Restelli. Tree-based fitted q-iteration for multi-objective markov decision problems. In *The 2012 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2012.

Xi Chen, Ali Ghadirzadeh, Mårten Björkman, and Patric Jensfelt. Meta-learning for multi-objective reinforcement learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 977–983. IEEE, 2019.

Eli Friedman and Fred Fontaine. Generalizing across multi-objective reward functions in deep reinforcement learning. *arXiv preprint arXiv:1809.06364*, 2018.

Justin Fu, John Co-Reyes, and Sergey Levine. Ex2: Exploration with exemplar models for deep reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Paul Glasserman and Xingbo Xu. Robust risk measurement and model risk. *Quantitative Finance*, 14(1):29–58, 2014.

Luigi Guiso, Paola Sapienza, and Luigi Zingales. Time varying risk aversion. *Journal of Financial Economics*, 128(3):403–421, 2018.

Abdullah Konak, David W. Coit, and Alice E. Smith. Multi-objective optimization using genetic algorithms: A tutorial, 2006.

Kaiwen Li, Tao Zhang, and Rui Wang. Deep reinforcement learning for multiobjective optimization. *IEEE transactions on cybernetics*, 51(6):3103–3114, 2020.

JiGuan G. Lin. On min-norm and min-max methods of multi-objective optimization. *Math. Program.*, 103(1):1–33, may 2005. ISSN 0025-5610.

Francisco S Melo. Convergence of q-learning: A simple proof. *Institute Of Systems and Robotics, Tech. Rep*, pages 1–4, 2001.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

Hossam Mossalam, Yannis M Assael, Diederik M Roijers, and Shimon Whiteson. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707*, 2016.

Sriraam Natarajan and Prasad Tadepalli. Dynamic preferences in multi-criteria reinforcement learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 601–608, 2005.

Simone Parisi, Matteo Pirotta, Nicola Smacchia, Luca Bascetta, and Marcello Restelli. Policy gradient approaches for multi-objective sequential decision making. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 2323–2330. IEEE, 2014.

Matteo Pirotta, Simone Parisi, and Marcello Restelli. Multi-objective reinforcement learning with continuous pareto frontier approximation. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015a.

Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 100(2):255–283, 2015b.

Mathieu Reymond and Ann Nowé. Pareto-dqn: Approximating the pareto front in complex multi-objective decision problems. In *Proceedings of the adaptive and learning agents workshop (ALA-19) at AAMAS*, 2019.

Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In *International conference on machine learning*, pages 4344–4353. PMLR, 2018.

Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems*, 8, 1995.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Sebastian B Thrun. Efficient exploration in reinforcement learning. 1992.

Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, and Evan Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine learning*, 84(1):51–80, 2011.

Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. Scalarized multi-objective reinforcement learning: Novel design techniques. In *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 191–199. IEEE, 2013.

Harm Van Seijen, Mehdi Fatemi, Joshua Romoff, Romain Laroche, Tavian Barnes, and Jeffrey Tsang. Hybrid reward architecture for reinforcement learning. *Advances in Neural Information Processing Systems*, 30, 2017.

Markus Wulfmeier, Abbas Abdolmaleki, Roland Hafner, Jost Tobias Springenberg, Michael Neunert, Tim Hertweck, Thomas Lampe, Noah Siegel, Nicolas Heess, and Martin Riedmiller. Regularized hierarchical policies for compositional transfer in robotics. 2019.

Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. *Advances in Neural Information Processing Systems*, 32, 2019.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Marcela Zuluaga, Andreas Krause, and Markus Püschel. ε-pal: an active learning approach to the multi-objective optimization problem. *The Journal of Machine Learning Research*, 17(1):3619–3650, 2016.

Appendix A. Ablation Study

In this Section we provide an ablation study and explore the effects of exploration with exemplar models (Fu et al., 2017) and adaptive task weights using wLSE (Buet-Golfouse and Utyagulov, 2022). The ablation study is carried out by understanding the effects of exemplar exploration and wLSE on the model's performance.

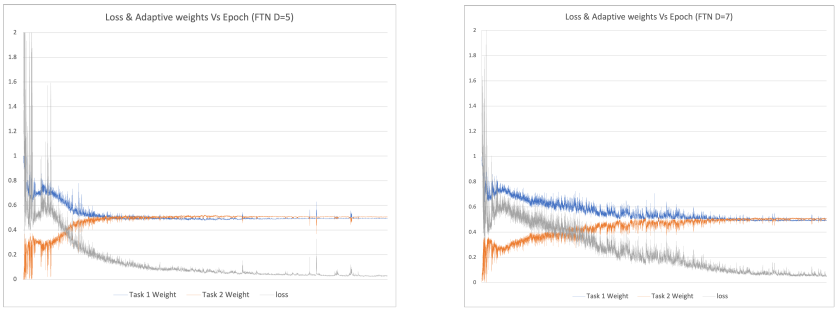

Figure 4: Convergence plots for Fruit Tree Network under dynamic preferences setting, Task 1: multi-objective Q-network loss, Task 2: policy Q-network loss

We evaluate the ability of the model to retrieve the CCS on two environments, Fruit Tree and Deep Sea Treasure and consider four scenarios: (1) Replacing wLSE with weighted average while computing the loss of the overall network (without wLSE), (2) Removing exemplar exploration, (3) replacing wLSE with weighted average and using only ϵ-greedy exploration, (4) using both wLSE and exemplar exploration. The coverage ratio results are provided in Tables 8-9 and the convergence plot under (4) in figure 4. In environments with smaller state space, FTN ($depth = 5$) and less objectives, DST, there is no scalability requirements while in complex scenarios with high dimensionality surrogate state spaces, the performance suffers. This is evident in FTN ($depth = 6$ and 7) where we see a boost in F1 score due to both efficient exploration and adaptive task weights.

Table 8: Fruit Tree Network: The networks are trained on 3000 episodes and the coverage ratio is calculated on a random sample of 2000 preferences over 10 trails

Tree Depth	Without wLSE	Without Exemplar	Without wLSE and Exemplar	with Both
5	1.0000 ±0.000	1.0000 ±0.000	1.0000 ±0.000	1.0000 ±0.000
6	0.9421 ±0.007	0.9201 ±0.005	0.9104 ±0.005	0.9912 ±0.009
7	0.7057 ±0.015	0.6181 ±0.010	0.7726 ±0.013	0.8326 ±0.020

Table 9: Deep Sea Treasure: The networks are trained on 2000 episodes and the coverage ratio is calculated on a random sample of 2000 preferences over 10 trails

Without wLSE	Without Exemplar	Without wLSE and Exemplar	with Both
1.0000 ±0.000	1.0000 ±0.000	0.8720 ±0.002	1.0000 ±0.000