Provably Efficient Representation Selection in Low-rank Markov Decision Processes: From Online to Offline RL

Abstract

The success of deep reinforcement learning (DRL) lies in its ability to learn a representation that is well-suited for the exploration and exploitation task. To understand how the choice of representation can improve the efficiency of reinforcement learning (RL), we study representation selection for a class of low-rank Markov Decision Processes (MDPs) where the transition kernel can be represented in a bilinear form. We propose an efficient algorithm, called ReLEX, for representation learning in both online and offline RL. Specifically, we show that the online version of ReLEX, called ReLEX-UCB, always performs no worse than the state-of-the-art algorithm without representation selection, and achieves a strictly better constant regret if the representation function class has a "coverage" property over the entire state-action space. For the offline counterpart, ReLEX-LCB, we show that the algorithm can find the optimal policy if the representation class can cover the state-action space and achieves gap-dependent sample complexity. This is the first result with constant sample complexity for representation learning in offline RL.

1 INTRODUCTION

Reinforcement Learning (RL) has achieved impressive results in game-playing [Mnih et al., 2013], robotics [Kober et al., 2013], and many other tasks. However, most current RL tasks are challenging due to large state-action spaces that make traditional tabular methods intractable. Instead, function approximation methods can be applied to tackle this challenge. In this scheme, the state-action pairs are compressed to provide some compact *representations* that leverage the underlying structure in the MDP, and therefore allow the algorithm to generalize to unseen states.

In modern approaches, deep neural networks are often used as feature extractors to generate these representations. Since different feature extractors powered by different pretrained neural networks can be used, multiple valid representations are generated to encode the same state-action pair. However, how to select the *best* representation for different scenarios is not well addressed in the literature. Nonetheless, this task is crucial in many applications such as robotics, where a robot is usually equipped with different types of sensors working through different physical phenomena [de Bruin et al., 2018], like accelerometers, magnetic sensors, or laser sensors. These sensors estimate the current state of the robot and provide a representation of the current state as the output. However, the accuracy and robustness of these sensors vary in different states. Thus an intelligent system should utilize the most accurate and robust sensor in different states to achieve the best performance.

For online reinforcement learning, existing works on representation learning [Jiang et al., 2017, Agarwal et al., 2020, Modi et al., 2021, Uehara et al., 2021, Sun et al., 2019, Du et al., 2021] often assumed that the transition dynamic can be represented as a linear function of an unknown representation, and they proposed algorithms to learn a single representation with provable sample complexity guarantees. They do not consider the possibility of using different representations for different scenarios (i.e., state-action pairs). On the other hand, for offline reinforcement learning, representation learning is much less studied, with only a few notable exceptions [Uehara et al., 2021, Zhang et al., 2022]. Nevertheless, neither of these works considers selecting different representations for different scenarios.

Based on the above motivation, we are interested in the following research question:

Can selecting a good representation improve sample efficiency in (online and offline) RL?

Submitted to the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023). **To be used for reviewing only**.

In this paper, we answer the above question affirmatively for a class of low-rank Markov Decision Processes (MDPs) named bilinear MDP [Yang and Wang, 2020], where the transition kernel $\mathbb{P}(s'|s,a)$ can be written as a bilinear form of a known feature map $\boldsymbol{\phi}(s,a)$, unknown matrix $\mathbf{M}^*$, and known feature map $\boldsymbol{\psi}(s')$. Our goal is to select the best representation $\boldsymbol{\phi}(s,a)$ from a finite representation class Φ for different (s,a) such that the resulting RL algorithm outperforms that using a single representation for all state-action pairs. For both online and offline reinforcement learning, we propose an algorithm called ReLEX, which can select the best representation in a representation function class in different scenarios. The key idea behind the representation selection is to choose the representation which gives the smallest optimistic Q-value function[1]. Our contributions are summarized as follows:

- In the context of online reinforcement learning, we propose a novel algorithm named ReLEX-UCB, which capitalizes on the benefits of representation selection. Our results show that ReLEX-UCB performs as well as the state-of-the-art algorithms that do not select representations, and attains a strictly superior regret bound when the representation function class has good coverage for all state-action pairs under the optimal policy.
- For offline reinforcement learning, we introduce ReLEX-LCB as a counterpart to ReLEX-UCB for the online setting. We demonstrate that ReLEX-LCB is capable of identifying the optimal policy with gap-dependent sample complexity of the offline data. Furthermore, when the representation function class satisfies certain coverage assumptions under the behavior policy, our algorithm enjoys a constant sample complexity, which represents a novel contribution to this line of research.
- To validate the effectiveness of representation selection and the superiority of our algorithms, we conduct empirical studies on various MDPs with different representation functions. Our experimental results demonstrate that both ReLEX-UCB and ReLEX-LCB outperform any single representation function in the respective settings, thus confirming the power of representation selection and the advantages of our proposed algorithms.

Notation. Scalars and constants are denoted by lower and upper case letters, respectively. Vectors are denoted by lower case boldface letters $\mathbf{x}$, and matrices by upper case boldface letters $\mathbf{A}$. We denote by $[k]$ the set $\{1,2,\cdots,k\}$ for positive integers k. For two non-negative sequence $\{a_n\},\{b_n\}$, $a_n=\mathcal{O}(b_n)$ means that there exists a positive constant C such that $a_n \le Cb_n$, and we use $\widetilde{\mathcal{O}}(\cdot)$ to hide the $\log$ factor in $\mathcal{O}(\cdot)$ except for the episode number k. We denote by $\|\cdot\|_2$ the Euclidean norm of vectors and the spectral norm of matrices and by $\|\cdot\|_{\mathrm{F}}$ the Frobenius norm of a matrix. We denote the Loewner ordering between two symmetric matrices as $\mathbf{A}\succeq\mathbf{B}$ if $\mathbf{A}-\mathbf{B}\succeq\mathbf{0}$. For a vector $\mathbf{x}\in\mathbb{R}^d$, we denote by $\mathbf{x}_{[i]}$ the i-th element of $\mathbf{x}$, for a matrix $\mathbf{A}\in\mathbb{R}^{d\times d}$, we denote by $\mathbf{A}_{[ii]}$ the i-th diagonal element. For any symmetric matrix $\mathbf{A}\in\mathbb{R}^{d\times d}$ and vector $\mathbf{x}\in\mathbb{R}^d$, we denote $\|\mathbf{x}\|_{\mathbf{A}}=\sqrt{\mathbf{x}^\top\mathbf{A}\mathbf{x}}$.

[1]For offline RL, our algorithm chooses the representation which gives the largest pessimistic Q-value function.

2 RELATED WORKS

In this section, we discuss related works on representation learning and selection in both online and offline RL. Additional related works are discussed in Appendix 1.

Learning good representations in reinforcement learning has a long history. One of the earliest methods for aggregating different states and generating a compressed representation for those states is state aggregation [Michael and Jordan, 1995, Dean and Givan, 1997, Ravindran and Barto, 2002, Abel et al., 2016]. In deep RL, deep neural networks have been used to learn good representations in different settings [Diuk et al., 2008, Stooke et al., 2020, Yang et al., 2020]. Several theoretical works [Du et al., 2019, Misra et al., 2020, Foster et al., 2020] have studied the Block MDP where the dynamics are governed by a discrete latent state space and proposed algorithms based on decoding the latent state space from the observations. Du et al. [2020] showed that having a good approximate representation for the Q-function, transition kernel, or optimal Q-function is not sufficient for efficient learning, and can still have an exponential sample complexity unless the quality of the approximation is above a certain threshold. In the linear function approximation setting, several representation learning algorithms have been proposed. For example, Jiang et al. [2017] proposed a model-free algorithm called OLIVE, which can learn the correct representation from a representation function class (in the realizable setting). Modi et al. [2021] improved the OLIVE algorithm by proposing the MOFFLE algorithm, which is computationally efficient. On the other hand, Agarwal et al. [2020] proposed a model-based algorithm, FLAMBE, which can find the correct representation from the representation function class. Uehara et al. [2021] improved FLAMBE by combining the maximum likelihood estimator and optimistic estimation (resp. pessimistic estimation) for representation learning in online RL (resp. offline RL). Some recent works [Qiu et al., 2022, Zhang et al., 2022] have used contrastive learning instead of the maximum likelihood estimator in Agarwal et al. [2020], Uehara et al. [2021] to obtain more practical algorithms.

All the aforementioned works focus on learning the "correct" representation, which can well approximate the underlying transition kernel. In contrast, we pursue a different objective, which is to select a good representation adaptively for different state-action pairs from a class of correct representations, which can potentially lead to better performance. To achieve this objective, Papini et al. [2021b] proposed an algorithm, LEADER, which leverages good representations

in linear contextual bandits. Independent of our work, Papini et al. [2021a] extended the representation selection in linear contextual bandits to linear MDPs [Jin et al., 2020]. The differences between their work and ours are as follows. First, they considered the linear MDP setting, which yields a linear dependence on the size of the representation class (i.e., $|\Phi|$) in their regret, while we studied a special linear MDP called bilinear MDP [Yang and Wang, 2020], which enjoys a logarithmic dependency on $|\Phi|$ (i.e., $\log(|\Phi|)$) in our regret. Second, we also consider representation learning in offline RL, which, to our knowledge, has not been considered before in the literature. Compared to previous results on representation learning in offline RL [Zhang et al., 2022, Uehara et al., 2021], we provide the first gap-dependent sample complexity

3 PRELIMINARIES

We consider time-inhomogeneous episodic Markov Decision Processes (MDP), denoted by $\mathcal{M}(\mathcal{S}, \mathcal{A}, H, r_h h = 1^H, \mathbb{P}_h h = 1^H)$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the finite action space, H is the length of each episode, $r_h : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ is the reward function at step h, and $\mathbb{P}_h(s'|s, a)$ denotes the probability for state s to transition to state s' with action a at step h. We further assume that the initial state s_1 is randomly sampled from a distribution μ.

Given the MDP, we consider a deterministic policy $\pi = \{\pi_h\}_{h=1}^H$ as a sequence of functions where $\pi_h : \mathcal{S} \mapsto \mathcal{A}$ maps a state s to an action a. For each state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$ at time-step h, given the policy π, we denote the Q-function and value function as follows:

$$Q_h^\pi(s, a) = r_h(s, a) + \mathbb{E}\left[\sum_{h'=h+1}^{H} r_{h'}(s_{h'}, \pi_{h'}(s_{h'}))\right],$$
$$V_h^\pi(s) = Q_h^\pi(s, \pi_h(s)),$$

where $s_h = s, a_h = a$ and for all $h' \in [h, H]$, the distribution of $s_{h'+1}$ is given by $\mathbb{P}_{h'}(s_{h'}|s, a)$. Both $Q_h^\pi(s, a)$ and $V_h^\pi(s)$ are bounded in $[0, H]$ by definition. We further define the optimal value function as $V^h(s) := \sup_\pi V_h^\pi(s)$ and the optimal Q-function as $Q^h(s, a) := \sup_\pi Q_h^\pi(s, a)$. The optimal policy is denoted by $\pi_h^(s) := \operatorname{argmax}_\pi V_h^\pi(s)$, and we assume the optimal policy function π^* is unique.

For simplicity, we define $[\mathbb{P}_h V](s, a) = \mathbb{E}_{s' \sim \mathbb{P}_h(s'|s,a)} V(s')$ for any function $V : \mathcal{S} \mapsto \mathbb{R}$. With this notation, we have the following Bellman equation, as well as the Bellman optimality equation:

$$Q_h^\pi(s, a) = r_h(s, a) + [\mathbb{P}_h V_{h+1}^\pi](s, a),$$
$$Q_h^*(s, a) = r_h(s, a) + [\mathbb{P}_h V_{h+1}^*](s, a), \quad (3.1)$$

where V_{H+1}^* and V_{H+1}^π are set to be zero for any state s and policy π.

We will focus on learning the structure of the MDP in an online manner. The algorithm is designed to run for K episodes, where for each episode $k \in [K]$, the first step is to determine a policy $\pi^k = \{\pi_h^k\}_{h=1}^H$ based on the knowledge collected from the environment. The agent then follows the policy and the dynamics of the MDP. Specifically, at each step $h \in [H]$, the agent observes the state s_h^k, selects an action a_h^k using the policy π_h^k, transitions to the next state s_{h+1}^k generated by the MDP, and receives the reward r_{h+1}^k.

We define the cumulative regret for the first K episodes as $\text{Regret}(K) = \sum_{k=1}^K V_1^(s_1^k) - V_1^{\pi^k}(s_1^k)$, where $V_1^*(s_1^k)$ is the optimal value of the initial state in episode k and $V_1^{\pi^k}(s_1^k)$ is the value of the initial state in episode k under the policy π^k.

The aim of this paper is to establish a problem-dependent regret bound. To achieve this goal, we require the assumption of a strictly positive minimal sub-optimality gap [Simchowitz and Jamieson, 2019, Yang et al., 2021, He et al., 2021]. This assumption ensures that the difference between the value of the optimal policy and the value of any other policy is not too small, which is essential for proving the regret bound.

Assumption 3.1. We have $\text{gap}_{\min} > 0$, where

$$\text{gap}_h(s, a) := V_h^*(s) - Q_h^*(s, a),$$
$$\text{gap}_{\min} := \inf_{h,s,a} \left\{\text{gap}_h(s, a) : \text{gap}_h(s, a) \neq 0\right\}. \quad (3.2)$$

We consider the bilinear MDPs in Yang and Wang [2020], where the probability transition kernel is a bi-linear function of the feature vectors.

Definition 3.2 (Bilinear MDPs, Yang and Wang 2020)**.** For each state-action-state triple $(s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, vectors $\boldsymbol{\phi}(s, a) \in \mathbb{R}^d, \boldsymbol{\psi}(s') \in \mathbb{R}^{d'}$ are known as the feature vectors. There exists an unknown matrix $\mathbf{M}_h^* \in \mathbb{R}^{d \times d'}$ for all $h \in [H]$ such that $\mathbb{P}_h(s'|s, a) = \boldsymbol{\phi}^\top(s, a)\mathbf{M}_h^* \boldsymbol{\psi}(s')$. We denote $\mathbf{K}_{\boldsymbol{\psi}} = \sum_{s \in \mathcal{S}} \boldsymbol{\psi}(s)\boldsymbol{\psi}^\top(s)$ which is assumed to be invertible. Let $\boldsymbol{\Psi} = (\boldsymbol{\psi}(s_1), \boldsymbol{\psi}(s_2), \cdots, \boldsymbol{\psi}(s_{|\mathcal{S}|}))^\top \in \mathbb{R}^{|\mathcal{S}| \times d'}$ be the matrix of all $\boldsymbol{\psi}$ features. We assume that for all $h \in [H]$, $\|\mathbf{M}_h^*\|_F^2 \leq C_{\mathbf{M}} d$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, $\|\boldsymbol{\phi}(s, a)\|_2^2 \leq C_{\boldsymbol{\phi}} d$, and for all $\mathbf{v} \in \mathbb{R}^{|\mathcal{S}|}$, $\|\boldsymbol{\Psi}^\top \mathbf{v}\|_2 \leq C_{\boldsymbol{\psi}} \|\mathbf{v}\|_\infty$ and $\|\boldsymbol{\Psi}\mathbf{K}_{\boldsymbol{\psi}}^{-1}\|_{2,\infty} \leq C'_{\boldsymbol{\psi}}$, where $C_{\mathbf{M}}, C_{\boldsymbol{\phi}}, C_{\boldsymbol{\psi}}$ and $C'_{\boldsymbol{\psi}}$ are all positive constants.

In this work, we focus on the bilinear MDP, which is a specific case of the low-rank MDP or linear MDP. In the low-rank MDP framework [Yang and Wang, 2019, Jin et al., 2020], the transition kernel is assumed to be a bilinear function of the state-action feature vector and an unknown measure $\boldsymbol{\theta}_h(s')$ of dimension d, i.e., $\mathbb{P}_h(s'|s, a) = \langle \boldsymbol{\phi}(s, a), \boldsymbol{\theta}_h(s') \rangle$. In our approach, we model $\boldsymbol{\theta}_h(s')$ as a product of an unknown matrix $\mathbf{M}_h^*$ and a feature vector $\boldsymbol{\psi}(s')$. In contrast, in the linear MDP, the reward function is

assumed to be a linear function of the state-action feature vector $\boldsymbol{\phi}(s,a)$, whereas we assume it is known to simplify the presentation. As noted by Yang and Wang [2020], we can replace this assumption with a linear function of the representation $\boldsymbol{\phi}(s,a)$ and add an optimistic reward function estimation step similar to LinUCB [Chu et al., 2011] without significantly altering the analysis.

Given the linear function representation of the MDP, we aim to learn a good representation $\boldsymbol{\phi}(s,a)$ for different state-action pairs in the representation function class Φ, in both online and offline settings. To this end, we introduce the definition of an *admissible* representation function class.

Definition 3.3 (Admissible Function Class)**.** A representation function class Φ is admissible if every $\boldsymbol{\phi} \in \Phi$ satisfies Definition 3.2 with a different dimension $d_{\boldsymbol{\phi}}$, a different parameter $\mathbf{M}^*_{h,\boldsymbol{\phi}}$, a different constant $C_{\boldsymbol{\phi}}$, and the same context $\boldsymbol{\psi}(s')$. In other words, for any representation function $\boldsymbol{\phi} \in \Phi$, the same transition kernel can be represented as $\mathbb{P}(s'|s,a)h = \boldsymbol{\phi}^\top(s,a)\mathbf{M}h, \boldsymbol{\phi}^*\boldsymbol{\psi}(s')$.

Remark 3.4. Definition 3.3 suggests that the same transition kernel can be represented in different ways, which is quite common in practice. For example, one can always represent a bilinear MDP with finite state and action spaces by a tabular MDP, which can be further represented by another bilinear MDP with $d_{\boldsymbol{\phi}} = |\mathcal{S}| \times |\mathcal{A}|$. However, different representations $\boldsymbol{\phi}$ may have different learning complexities. For instance, the linear representations with a lower dimension $d_{\boldsymbol{\phi}}$ are easier to learn than the tabular representation with $d_{\boldsymbol{\phi}} = |\mathcal{S}| \times |\mathcal{A}|$. Thus, our goal is to select a good representation from the admissible function class for different state-action pairs.

Remark 3.5. Although one can also consider learning the representation function tuple $(\boldsymbol{\phi}, \boldsymbol{\psi}) \in \Phi \times \Psi$ simultaneously, there is no difference compared with assuming the $\boldsymbol{\psi}$ function is fixed and known in terms of the algorithm and the analysis. This is because the Q-function is a linear function of $\boldsymbol{\phi}$ (see (4.2)), and the confidence radius of the estimated Q-function only depends on $\boldsymbol{\phi}$ (see (4.3)). Therefore, we only select the representation of $\boldsymbol{\phi}$ instead of both $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ without loss of generality.

4 REPRESENTATION SELECTION FOR ONLINE RL

4.1 RELEX-UCB ALGORITHM

We present the *Representation seLection for EXploration and EXploitation* with upper confidence bound (ReLEX-UCB) algorithm for selecting a good representation from a finite representation function class Φ for different state-action pairs. The algorithm, shown in Algorithm 1, maintains a different model parameter estimate for each individual representation $\boldsymbol{\phi} \in \Phi$. Under Definition 3.2, we have the following property for each representation $\boldsymbol{\phi}$:

$$\begin{aligned}\left[\mathbb{P}_h\boldsymbol{\psi}(\cdot)^\top\mathbf{K}_{\boldsymbol{\psi}}^{-1}\right](s,a) &= \sum_{s'\in\mathcal{S}}\mathbb{P}_h(s'|s,a)\boldsymbol{\psi}^\top(s')\mathbf{K}_{\boldsymbol{\psi}}^{-1}\\ &= \sum_{s'\in\mathcal{S}}\boldsymbol{\phi}^\top(s,a)\mathbf{M}^*_{h,\boldsymbol{\phi}}\boldsymbol{\psi}(s')\boldsymbol{\psi}^\top(s')\mathbf{K}_{\boldsymbol{\psi}}^{-1}\\ &= \boldsymbol{\phi}^\top(s,a)\mathbf{M}^*_{h,\boldsymbol{\phi}}, \end{aligned} \tag{4.1}$$

where the last equality uses the fact that $\mathbf{K}_{\boldsymbol{\psi}} = \sum_{s'\in\mathcal{S}}\boldsymbol{\psi}(s')\boldsymbol{\psi}^\top(s')$. Equation (4.1) suggests that we can build $\mathbf{M}^k_{h,\boldsymbol{\phi}}$, the estimate of $\mathbf{M}^*_{h,\boldsymbol{\phi}}$, as the solution to the ridge regression problem analytically, given the sampled triples $\{s^j_h, a^j_h, s^j_{h+1}\}_{j=1}^{k-1}$ in Line 6 of Algorithm 1.

Remark 4.1. The computation of $\mathbf{K}_{\boldsymbol{\psi}}$ requires only one pass of the state space since it does not depend on the round k or the representation $\boldsymbol{\phi}$. Thus, it is not computationally expensive and would not be a bottleneck in the algorithm's computational complexity. Additionally, when the state space is infinite, $\mathbf{K}_{\boldsymbol{\psi}}$ can be efficiently approximated using Monte Carlo integration techniques, as demonstrated in prior works such as Zhou et al. [2021] and Yang and Wang [2020].

With the estimate $\mathbf{M}^k_{h,\boldsymbol{\phi}}$, Algorithm 1 recursively estimates the Q-function starting from $Q^k_{H+1} = 0$. The Q-function at step h can be deduced as $Q^k_{h,\boldsymbol{\phi}}(s,a) = r(s,a) + [\mathbb{P}_h V^k_{h+1}](s,a)$, where V^k_{h+1} is the estimate of the value function at step $h+1$. Using the Bellman equation (3.1), we can further write $Q^k_{h,\boldsymbol{\phi}}(s,a)$ as

$$Q^k_{h,\boldsymbol{\phi}}(s,a) = r(s,a) + \sum_{s'\in\mathcal{S}}\boldsymbol{\phi}^\top(s,a)\mathbf{M}^k_{h,\boldsymbol{\phi}}\boldsymbol{\psi}(s')V^k_{h+1}(s'). \tag{4.2}$$

To construct an optimistic estimation of the Q-function, we follow the approach proposed by Yang and Wang [2020] and add an optimism bonus term to the right-hand side of (4.2). The optimism bonus is defined as $C_{\boldsymbol{\psi}}H\sqrt{\beta_{k,\boldsymbol{\phi}}}\|\boldsymbol{\phi}(s,a)\|(\mathbf{U}^k_{h,\boldsymbol{\phi}})^{-1}$, where $C_{\boldsymbol{\psi}}$ and $\beta_{k,\boldsymbol{\phi}}$ are user-defined hyperparameters, and $\mathbf{U}^k_{h,\boldsymbol{\phi}}$ is the covariance matrix calculated in Line 7. This results in the following optimistic estimation of the Q-function:

$$\begin{aligned}Q^k_{h,\boldsymbol{\phi}}(s,a) = r(s,a) &+ \sum_{s'\in\mathcal{S}}\boldsymbol{\phi}^\top(s,a)\mathbf{M}^k_{h,\boldsymbol{\phi}}\boldsymbol{\psi}(s')V^k_{h+1}(s')\\ &+ C_{\boldsymbol{\psi}}H\sqrt{\beta_{k,\boldsymbol{\phi}}}|\boldsymbol{\phi}(s,a)|(\mathbf{U}h,\boldsymbol{\phi}^k)^{-1}. \end{aligned} \tag{4.3}$$

By following the standard analysis for optimistic estimation [Abbasi-Yadkori et al., 2011], it can be shown that $Q^k_{h,\boldsymbol{\phi}}(s,a)$ is an upper confidence bound for $Q^{\pi^k}_h(s,a)$ for each representation $\boldsymbol{\phi} \in \Phi$, according to (4.3). In other words, $Q^k_{h,\phi}(s,a) \geq Q^*_h(s,a)$. In Line 10, the algorithm selects the representation with the smallest optimistic estimation, which should be considered as the tightest optimistic estimation given the current covariance matrix $\mathbf{U}^k_{h,\boldsymbol{\phi}}$.

Algorithm 1 Online Representation seLection for EXploration and EXploitation with Upper Confidence Bound (ReLEX-UCB)

1: Initialize $Q^k_{H+1,\boldsymbol{\phi}}(s,a) = 0$ for all $(s,a,k,\boldsymbol{\phi})$
2: **for** episodes $k = 1, \dots, K$ **do**
3: Received the initial state s^k_1.
4: **for** step $h = H, \dots, 1$ **do**
5: **for** representation $\boldsymbol{\phi} \in \Phi$ **do**
6: $\mathbf{M}^k_{h,\boldsymbol{\phi}} = \operatorname{argmin}_{\mathbf{M}} \left(\sum_{j=1}^{k-1} \|\boldsymbol{\psi}^\top(s^k_{h+1})\mathbf{K}^{-1}_{\boldsymbol{\psi}} - \boldsymbol{\phi}^\top(s^k_h, a^k_h)\mathbf{M}\|_2^2 + \|\mathbf{M}\|_F^2 \right)$
7: $\mathbf{U}^k_{h,\boldsymbol{\phi}} = \mathbf{I} + \sum_{j=1}^{k-1} \boldsymbol{\phi}(s^k_h, a^k_h)\boldsymbol{\phi}^\top(s^k_h, a^k_h)$
8: Calculate $Q^k_{h,\boldsymbol{\phi}}(s,a)$ as (4.3)
9: **end for**
10: Set $Q^k_h(s,a) = \min_{\boldsymbol{\phi}\in\Phi}\{Q^k_{h,\boldsymbol{\phi}}(s,a)\}$,
11: Set $V^k_h(s) = \max\{0, \min\{\max_a Q^k_h(s,a), H\}\}$
12: **end for**
13: **for** step $h = 1, \dots, H$ **do**
14: Take action $a^k_h \leftarrow \operatorname{argmax}_a Q^k_h(s^k_h, a)$
15: Receive next state s^k_{h+1}
16: **end for**
17: **end for**

Alternatively, this step can be interpreted as selecting the representation with the minimal uncertainty, which is measured by $\|\boldsymbol{\phi}\|_{(\mathbf{U}^k_{h,\boldsymbol{\phi}})^{-1}}$. This approach is intuitive and ensures that the algorithm chooses the representation that provides the best possible estimate of the value function for the current state-action pair.

As a result, Line 10 selects different representations $\boldsymbol{\phi}$ for different state-action pairs implicitly by minimizing the optimistic Q-function, which results in a tighter optimistic estimation of the Q-function. The algorithm then executes the greedy policy and obtains the optimistic value function defined in Line 11.

Our algorithm offers a distinct advantage over Yang and Wang [2020] in that it enables the selection of different representations for different state-action pairs. This is in contrast to representation learning, which seeks a *universal* representation that works well for *all* state-action pairs. For instance, our algorithm can adaptively choose a representation that yields accurate value function estimates for certain state-action pairs, even if its performance is suboptimal for others. By doing so, our algorithm outperforms Yang and Wang [2020] which relies on a single representation for all state-action pairs. This demonstrates the benefits of representation selection in online RL.

4.2 CONSTANT REGRET BOUNDS

We present the regret bound for ReLEX-UCB, which demonstrates the advantage of representation selection in a rigorous way. To do so, we require the following assumption:

Assumption 4.2. Suppose that the representation function class Φ is admissible. For any $(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]$, there exists a representation $\boldsymbol{\phi} \in \Phi$ such that $\boldsymbol{\phi}(s,a) \in \operatorname{Im}(\boldsymbol{\Lambda}_{h,\boldsymbol{\phi}})$, where

$$\boldsymbol{\Lambda}_{h,\boldsymbol{\phi}} := \mathbb{E}_{d_{\pi^*}}\left[\phi(s_h, \pi^*_h(s_h))\phi^\top(s_h, \pi^*_h(s_h))\right]$$

with d_{π^*} representing the state visitation distribution induced by the optimal policy π^*. We also denote $\sigma_{h,\boldsymbol{\phi}}$ as the minimal non-zero eigenvalue of $\boldsymbol{\Lambda}_{h,\boldsymbol{\phi}}$ and $\sigma_{\boldsymbol{\phi}} = \min_{h\in[H]} \sigma_{h,\boldsymbol{\phi}}$.

Remark 4.3. Several related assumptions, known as *diversity assumptions*, have been proposed to lower bound the minimum eigenvalue of the term $\boldsymbol{\phi}\boldsymbol{\phi}^\top$. These assumptions are discussed in detail in Papini et al. [2021b]. Here, we extend the assumption from the linear bandit to the reinforcement learning setting, where the state distribution at time-step h is defined by the optimal policy. We note that a similar but stronger assumption, called 'uniformly excited features', is made by Wei et al. [2021] in the infinite time-horizon average reward MDP setting. There, they assume $\boldsymbol{\Lambda}$ is strictly positive definite for *all* possible policies π. In contrast, we only require $\boldsymbol{\Lambda}$ to be strictly positive for the distribution induced by the *optimal* policy, which is a weaker assumption. This implies that the states that rarely occur in the optimal policy do not significantly impact the quality of the representation.

Remark 4.4. We notice that a similar assumption called UniSoft-mixing, is made in Papini et al. [2021a], where they assume that for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, there exists a $\boldsymbol{\phi} \in \Phi$ such that $\boldsymbol{\phi}(s,a) \in \operatorname{span}\{\boldsymbol{\phi}(s, \pi^*(s)) : d_{\pi^*}(s) > 0\}$. The difference between our assumption and their assumption is that they filter out the states which are *almost surely* never visited by the optimal policy π^*. In contrast, we take the expectation with respect to $d_{\pi^*}(s)$ without explicitly filtering out the never-visited states.

Now we are ready to present the regret bound result.

Theorem 4.5. Under Assumptions 3.1 and 4.2, set $\beta_{k,\boldsymbol{\phi}} = c(C_{\mathbf{M}} + {C'_{\boldsymbol{\psi}}}^2)d_{\boldsymbol{\phi}} \log(kHC_{\boldsymbol{\phi}}|\Phi|/\delta)$ in Algorithm 1, where c is an absolute positive constant, then with probability at least $1 - 5\delta$, there exists a threshold

$$k^* = \max_{\boldsymbol{\phi}\in\Phi} \Big\{ \operatorname{poly}(d_{\boldsymbol{\phi}}, \sigma^{-1}_{\boldsymbol{\phi}}, H, \log(|\Phi|/\delta), \operatorname{gap}^{-1}_{\min}, C_{\boldsymbol{\phi}}, C_{\boldsymbol{\psi}}, C_{\mathbf{M}}, C'_{\boldsymbol{\psi}}) \Big\} \quad (4.4)$$

independent from episode number k. The regret for the first

k episodes is upper bounded by

$$\begin{aligned}\text{Regret}(k) \leq 2 + \min_{\boldsymbol{\phi}\in\Phi} \Bigg\{ & \frac{128 C_{\boldsymbol{\psi}}^2 H^5 d_{\boldsymbol{\phi}}^2 c (C_{\mathbf{M}} + {C'_{\boldsymbol{\psi}}}^2)}{\text{gap}_{\min}} \\ & \times \log\left(1 + C_{\boldsymbol{\phi}} \widetilde{k} d_{\boldsymbol{\phi}}\right) \log\left(\widetilde{k} H C_{\boldsymbol{\phi}} |\Phi|/\delta\right) \Bigg\} \\ & + \frac{96 H^4 \log\left(2\widetilde{k}(1 + \log(H \text{gap}_{\min}^{-1}))|\Phi|/\delta\right)}{\text{gap}_{\min}} \\ & + \frac{16 H^2 \log\left(\left(1 + \log\left(H\widetilde{k}\right)\right)\widetilde{k}^2 |\Phi|/\delta\right)}{3},\end{aligned}$$

where we denote $\widetilde{k} := \min\{k, k^*\}$.

Remark 4.6. The regret bound exhibits a phase transition as the episode number k increases. When $k \leq k^*$, the regret is upper bounded by $\widetilde{\mathcal{O}}(d^2 H^5 \log(k) \text{gap}_{\min}^{-1})$, which is exactly the logarithmic regret bound (given by Lemma 3.3 in Appendix). However, when $k \geq k^*$, the regret bound becomes $\widetilde{\mathcal{O}}(d^2 H^5 \log(k^*) \text{gap}_{\min}^{-1})$. Since k^* is independent of k (as shown in (4.4)), the regret bound turns into a problem-dependent constant regret bound that no longer grows as the total number of episodes k increases. This result aligns with our intuition: once we have a fixed, strictly positive sub-optimality gap, the regret might initially increase over the first few episodes. However, once the agent collects enough data, it can learn the environment well and will no longer incur any additional regret.

Remark 4.7. If Assumption 4.2 does not hold, then $k^* = \infty$, and our regret bound degenerates to the gap-dependent regret bound. Similar bounds have been proved in He et al. [2021] for both linear MDPs and linear mixture MDPs. Our bound has the same dependency on H, $\text{gap}_{\min}$, and episode number k as the bounds in He et al. [2021]. However, in terms of d, our dependency is $\mathcal{O}(d^2)$, while the dependency is $\mathcal{O}(d^3)$ for linear MDPs in the LSVI-UCB algorithm [Jin et al., 2020]. This difference arises from estimating the MDP parameter $\mathbf{M}^*_{h,\boldsymbol{\phi}}$, which is similar to that in the UCRL-VTR algorithm [Ayoub et al., 2020] for learning linear mixture MDPs. Furthermore, since our regret bound minimizes over all $\boldsymbol{\phi} \in \Phi$, the performance of ReLEX-UCB is always competitive with the best one using any single representation $\boldsymbol{\phi}$ in that function class, ignoring the logarithmic terms.

Remark 4.8. We note that our regret bound includes an additional $\log(|\Phi|)$ factor, which reflects the cost of representation selection to guarantee that all $|\Phi|$ regressions can be learned well by the union bound. This term is caused by the worst-case scenario and may be eliminated in practice by considering the average-case scenario instead. By doing so, we can potentially reduce the impact of the $\log(|\Phi|)$ factor on the regret bound. Additionally, it's worth noting that this dependency on $|\Phi|$ is better than the one in the regret bound of Papini et al. [2021a], which has a $|\Phi|$ factor. The reason for the better dependency in our result is that the bilinear MDP structure we consider is simpler than the linear MDP structure considered in Papini et al. [2021a]. When applying our algorithm to linear MDPs, we still need a $|\Phi|$ factor to cover the value function class, which degenerates to the result in Papini et al. [2021a]. Furthermore, the $\log(|\Phi|)$ dependency allows us to extend our result to some infinite representation function classes with bounded statistical complexity [Agarwal et al., 2020].

Remark 4.9. When $|\Phi| = 1$, i.e., there is only one representation function, Assumption 4.2 provides a criterion for a 'good representation' and such a 'good representation' can improve the problem-dependent regret bound from $\mathcal{O}(\log(k))$ [He et al., 2021] to a constant regret bound.

5 REPRESENTATION SELECTION FOR OFFLINE RL

5.1 RELEX-LCB ALGORITHM

We present an offline version of ReLEX that selects a good representation based on the offline data generated from a behavior policy. In this version, the algorithm estimates the parameter and its covariance matrix for each representation function $\boldsymbol{\phi}$ in Lines 3 and 4 in Algorithm 2, using the offline data $\mathcal{D}_h$ for the h-th step, which consists of the triplet (s, a, s') as the state, action, and next-state, then the estimated $\mathbf{M}$ can be therefore written by

$$\begin{aligned}\mathbf{M}_{h,\boldsymbol{\phi}} = & \operatorname*{argmin}_{\mathbf{M}} \|\mathbf{M}\|_F^2 \\ & + \sum_{(s,a,s')\in\mathcal{D}_h} \|\boldsymbol{\psi}^\top(s')\mathbf{K}_{\boldsymbol{\psi}}^{-1} - \boldsymbol{\phi}^\top(s,a)\mathbf{M}\|_2^2\end{aligned} \tag{5.1}$$

The algorithm then provides a pessimistic estimation of the Q-function, following a similar method as (4.3) in Lines 9 and 10, which is widely used in offline reinforcement learning to provide a robust estimation for later planning. In detail, the estimated Q function is subtracted by a confidence radius Γ defined by

$$\Gamma h, \boldsymbol{\phi}(s,a) = C_{\boldsymbol{\psi}} H \sqrt{\beta_{\boldsymbol{\phi}} \boldsymbol{\phi}^\top(s,a) \mathbf{U}_{h,\boldsymbol{\phi}}^{-1} \boldsymbol{\phi}(s,a)} \tag{5.2}$$

and thus the estimated Q-function can be written as

$$\begin{aligned}Q_{h,\boldsymbol{\phi}}(s,a) = & r(s,a) + \sum_{s'\in\mathcal{S}} \boldsymbol{\phi}^\top(s,a)\mathbf{M}_{h,\boldsymbol{\phi}}\boldsymbol{\psi}(s')V_{h+1}(s') \\ & - \Gamma_{h,\boldsymbol{\phi}}(s,a).\end{aligned} \tag{5.3}$$

Unlike the online version, where a smaller estimation of Q is preferred, the offline version adopts a pessimistic estimation ($Q_{h,\boldsymbol{\phi}} \leq Q_h^*$), where a larger estimation is considered more accurate. Therefore, in Line 11, the algorithm selects the maximum Q-function over all representation functions $\boldsymbol{\phi}$, and in Line 13, it takes the greedy policy based on the

Algorithm 2 Offline Representation seLection for EXploration and EXploitation Lower Confidence Bound (ReLEX-LCB)

```
// offline training
for (h, φ) ∈ [H] × Φ do
    Calculate M_{h,φ} as of (5.1)
    Calculate U_{h,φ} = I + Σ_{(s,a,s')∈D_h} φ(s,a)φ^⊤(s,a)
end for
// offline planning
Initialize Q_{H+1,φ}(s,a) = 0 for all (s,a,φ)
for h = H, H−1, ···, 1 do
    Calculate Γ_{h,φ}(s,a) as of (5.2)
    Calculate Q_{h,φ}(s,a) as of (5.3)
    Set Q_h(s,a) = max_{φ∈Φ}{Q_{h,φ}(s,a)}
    Set V_h(s) = max{0, min{max_a Q_h(s,a), H}}
    Set π_h(s,a) = argmax_a Q_h(s,a)
end for
```

Output: Policy $\pi = \{\pi_h\}_{h=1}^H$

selected Q-function from the offline training. Similar to the online version, ReLEX-LCB selects different representation functions for different state-action pairs instead of a single representation for the entire environment, thereby leveraging the advantage of different representation functions to provide a good estimation for the underlying MDP.

5.2 GAP-DEPENDENT SAMPLE COMPLEXITY

In this section, we provide the sample complexity of Algorithm 2. Similarly to its online counterpart, we start with a coverage assumption for offline RL, which suggests that the representation function class Φ can provide a good representation for all possible state-action pairs in the offline training data.

Assumption 5.1. Suppose the representation function class Φ is admissible, and for any $(s, a, h) \in \mathcal{S} \times \mathcal{A} \times [H]$, there exists a representation function $\boldsymbol{\phi} \in \Phi$ such that

$$\boldsymbol{\phi}(s,a) \in \mathrm{Im}(\widetilde{\boldsymbol{\Lambda}}_{h,\boldsymbol{\phi}}),\ \widetilde{\boldsymbol{\Lambda}}_{h,\boldsymbol{\phi}} := \mathbb{E}_{d_h^{\widehat{\pi}}}[\boldsymbol{\phi}(s,a)\boldsymbol{\phi}(s,a)^\top],$$

where $d_h^{\widehat{\pi}}$ is the state-action visitation distribution in the offline dataset on step h induced by some behavior policy $\widehat{\pi}$ in the underlying MDP for the offline data. We denote the minimal non-zero eigenvalue of $\widetilde{\boldsymbol{\Lambda}}_{h,\boldsymbol{\phi}}$ as $\widetilde{\sigma}_{h,\boldsymbol{\phi}}$.

Remark 5.2. Similar assumptions have been made in the offline RL literature [Wang et al., 2020, Jin et al., 2021, Min et al., 2021, Uehara et al., 2021, Yin et al., 2022], which require that the offline dataset can provide good coverage of the entire state-action space. Notably, thanks to representation selection, we only require that the representations in the function class Φ can together cover the state-action space, rather than every single representation covering the state-action space perfectly. This relaxes existing assumptions by allowing every single representation to not provide perfect coverage. For example, it is possible to define two representations $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2$ such that each representation does not satisfy Assumption 5.1, but the representation function class $\Phi = \boldsymbol{\phi}_1, \boldsymbol{\phi}_2$ satisfies. For more details about this example, please refer to Appendix 2.2, or Appendix G in Papini et al. [2021a].

We also need the following assumption, which is standard in the literature.

Assumption 5.3. The trajectories in the offline dataset are i.i.d. sampled, i.e., different trajectories are generated by the same behavior policy $\widehat{\pi}$ independently.

Now we are ready to present the sample complexity result.

Theorem 5.4. Set $\beta_{\boldsymbol{\phi}} = Cd_{\boldsymbol{\phi}} \log(2KH|\Phi|/\delta)$ where C is an absolute positive constant, then with probability at least $1-\delta$, then under Assumptions 5.1 and 5.3, the sub-optimality of the policy π output by Algorithm 2 could be bounded by

$$\begin{aligned} &V_h^*(s) - V_h^\pi(s) \\ &\le 2C_{\boldsymbol{\psi}} H \sum_{h'=h}^{H} \mathbb{E}_{\pi^*}\Big[\min_{\boldsymbol{\phi}\in\Phi}\big\{\sqrt{\beta_{\boldsymbol{\phi}}}\|\boldsymbol{\phi}(s,a)\|_{\mathbf{U}_{h',\boldsymbol{\phi}}^{-1}}\big\}\big|s_h = s\Big]. \end{aligned} \tag{5.4}$$

Furthermore, under Assumptions 3.1, if the size of the offline dataset is greater than

$$\begin{aligned} K > \max_{\boldsymbol{\phi}\in\Phi, h\in[H]} \Bigg\{ &\frac{32C_{\boldsymbol{\phi}}^2 d_{\boldsymbol{\phi}}^2 \log(Hd_{\boldsymbol{\phi}}|\Phi|/\delta)}{\widetilde{\sigma}_{h,\boldsymbol{\phi}}^2} \\ &\times \left(1 + \frac{C_{\boldsymbol{\psi}}^2 H^4 \beta_{\boldsymbol{\phi}} C_{\boldsymbol{\phi}} \widetilde{\sigma}_{h,\boldsymbol{\phi}}}{4\mathrm{gap}_{\min}^2 C_{\boldsymbol{\phi}}^2 d_{\boldsymbol{\phi}} \log(Hd_{\boldsymbol{\phi}}|\Phi|/\delta)}\right)\Bigg\} \end{aligned}$$

then Algorithm 2 is guaranteed to output the optimal policy $\pi = \pi^*$.

Remark 5.5. Our error bound in (5.4) contains the $\min$ operator, which suggests that our result should be no worse than using any single representation, compared with the offline RL algorithm using a single representation [Jin et al., 2021, Yin et al., 2022]. Furthermore, to make the sub-optimality decrease at a $1/\sqrt{K}$ rate, Jin et al. [2021], Yin et al. [2022] require a 'uniform coverage' assumption, which suggests that the covariance matrix under the behavior policy can cover the entire state-action space. In sharp contrast, according to Assumption 5.1, our results only require the representations in the function class to together cover the state-action space, even if any single representation cannot.

Remark 5.6. Our 'gap-dependent sample complexity' is also aligned with the gap-dependent sample complexity for offline RL in the tabular setting under the condition $(P, \mathrm{gap}_{\min})$ in Wang et al. [2022]. In their setting, P stands for a uniform optimal policy coverage coefficient in the tabular MDP, which is analogous to our $\widetilde{\sigma}_{h,\boldsymbol{\phi}}^{-1}$ in the linear function approximation setting. Our result has the same inverse dependence on $\mathrm{gap}_{\min}$.

Table 1: Cumulative regret (mean $\pm$ dev.) after 5M episodes for ReLEX-UCB v.s. UC-MatrixRL and ϵ-greedy using a single representation

Alg. + Rep.	Cumulative regret
UC-MatrixRL + $\boldsymbol{\phi}$ (oracle)	2534.9 ± 26.6
UC-MatrixRL + $\boldsymbol{\phi}^{(1)}$	11459.5 ± 225.7
UC-MatrixRL + $\boldsymbol{\phi}^{(2)}$	13838.5 ± 266.2
ϵ-greedy + $\boldsymbol{\phi}$	15305.9 ± 245.7
ϵ-greedy + $\boldsymbol{\phi}^{(1)}$	15745.8 ± 408.0
ϵ-greedy + $\boldsymbol{\phi}^{(2)}$	15652.9 ± 471.2
ReLEX-UCB + $\{\boldsymbol{\phi}^{(1)}, \boldsymbol{\phi}^{(2)}\}$	$\mathbf{6765.0 \pm 146.6}$

6 EXPERIMENTS

6.1 ONLINE RL

To showcase the efficacy of representation selection by ReLEX-UCB, we conduct the following experiments on an environment with $|\mathcal{S}| = 20, |\mathcal{A}| = 3$, and $d = d' = 5$. We generate the feature functions $\boldsymbol{\phi} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}^d$ and $\boldsymbol{\psi} : \mathcal{S} \mapsto \mathbb{R}^{d'}$ such that for all $h \in [H]$, there exists a matrix $\mathbf{M}_h \in \mathbb{R}^{d \times d}$ where $\mathbb{P}_h(s'|s,a) = \boldsymbol{\phi}(s,a)^\top \mathbf{M}_h \boldsymbol{\psi}(s')$. The generated $\boldsymbol{\phi}$ satisfies Assumption 4.2. We set the reward function such that $r_H(s,a) \sim$ Bernoulli(0.5) and $r_h(s,a) = 0$ for all $h < H$, forcing the algorithm to learn the transition kernel in order to achieve good performance.

Furthermore, we generate two additional representations $\boldsymbol{\phi}^{(1)}$ and $\boldsymbol{\phi}^{(2)}$ such that neither $\boldsymbol{\phi}^{(1)}$ nor $\boldsymbol{\phi}^{(2)}$ satisfies Assumption 4.2, but their union $\Phi = \boldsymbol{\phi}^{(1)}, \boldsymbol{\phi}^{(2)}$ does. Appendix 2.1 contains a detailed definition of these representations.

We evaluated the performance of ReLEX-UCB using the feature map class $\Phi = \boldsymbol{\phi}^{(1)}, \boldsymbol{\phi}^{(2)}$ with an episode length of $K = 5,000,000$. We also reported the performance of UC-MatrixRL [Yang and Wang, 2020] and ϵ-greedy using the feature map $\boldsymbol{\phi}$, $\boldsymbol{\phi}^{(1)}$, and $\boldsymbol{\phi}^{(2)}$ separately.

We repeated the experiment on the same environment eight times and reported the mean and standard deviation of the cumulative regret in Table 1. Our experiment results showed that ReLEX-UCB outperformed both ϵ-greedy and UC-MatrixRL using $\boldsymbol{\phi}^{(1)}$ or $\boldsymbol{\phi}^{(2)}$, which verifies the effectiveness of representation selection. More results, including the figure of cumulative regret, are deferred to Appendix 2.4.1.

6.2 OFFLINE RL

In this subsection, we present experiments to demonstrate the performance of ReLEX-LCB. We use a setup similar to the online RL setting with one oracle representation $\boldsymbol{\phi}$ satisfying Assumption 5.1 and two representations $\boldsymbol{\phi}^{(1)}$ and $\boldsymbol{\phi}^{(2)}$. Neither of $\boldsymbol{\phi}^{(1)}$ nor $\boldsymbol{\phi}^{(2)}$ satisfies Assumption 5.1, but the union of these two representations satisfies the assumption. We collect $K = 500K$ episodes of offline trajectories using a fixed randomly-generated behavior policy and evaluate the sub-optimality of Algorithm 2 using different sizes of offline training data. The rest of the parameter settings are the same as in the online RL setting.

Table 2: Relative sub-optimality of ReLEX-LCB over 500K episodes

Representation	Final sub-optimality (mean $\pm$ dev.) $\times 10^{-3}$
$\boldsymbol{\phi}$ (oracle)	1.288 ± 0.807
$\boldsymbol{\phi}^{(1)}$	3.424 ± 1.455
$\boldsymbol{\phi}^{(2)}$	$3.336 \pm 1,624$
$\{\boldsymbol{\phi}^{(1)}, \boldsymbol{\phi}^{(2)}\}$	$\mathbf{1.292 \pm 0.806}$

We report the performance of Algorithm 2 using (1) the oracle representation $\boldsymbol{\phi}$, (2) the representation function class $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2$, (3) $\boldsymbol{\phi}_1$, and (4) $\boldsymbol{\phi}_2$, respectively. We use the relative sub-optimality over the initial policy, i.e., $(V_1^*(s) - V_1(s)^{\pi_k})/(V_1^*(s) - V_1(s)^{\pi_1})$ as a performance measure. We repeat the experiment 32 times and report the mean and standard deviation of the relative sub-optimality in Table 2.

We observe that by selecting over two imperfect representations, ReLEX-LCB can match the performance of the oracle algorithm using a single perfect representation, even if using the two representations separately leads to a larger ($\sim 2.5\times$) sub-optimality on the same offline data. More results, including the figures comparing the sub-optimality over different algorithms, are deferred to Appendix 2.4.1.

7 CONCLUSION AND FUTURE WORK

In this paper, we have explored representation selection for reinforcement learning by focusing on a special class [Yang and Wang, 2020] of low-rank MDPs [Yang and Wang, 2019, Jin et al., 2020]. Our proposed ReLEX algorithm has demonstrated the ability to improve performance in both online and offline RL settings. The promising theoretical and empirical results suggest that there is potential in combining our work with FLAMBE [Agarwal et al., 2020] or MOFFLE [Modi et al., 2021]. By integrating our approach with these methods that select the *correct* representations, we can further select the *good* representation from a class of *correct* representations. This may help in designing more practical, theory-backed representation learning algorithms for reinforcement learning.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, volume 11, pages 2312–2320, 2011.

David Abel, David Hershkowitz, and Michael Littman. Near optimal behavior via approximate state abstraction. In *International Conference on Machine Learning*, pages 2915–2923. PMLR, 2016.

Alekh Agarwal, Sham M Kakade, Akshay Krishnamurthy, and Wen Sun. Flambe: Structural complexity and representation learning of low rank mdps. *Advances in neural information processing systems*, 2020.

Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin Yang. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pages 463–474. PMLR, 2020.

Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214. JMLR Workshop and Conference Proceedings, 2011.

Tim de Bruin, Jens Kober, Karl Tuyls, and Robert Babuška. Integrating state representation learning into deep reinforcement learning. *IEEE Robotics and Automation Letters*, 3(3):1394–1401, 2018.

Thomas Dean and Robert Givan. Model minimization in markov decision processes. In *AAAI/IAAI*, pages 106–111, 1997.

Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 240–247, 2008.

Simon Du, Akshay Krishnamurthy, Nan Jiang, Alekh Agarwal, Miroslav Dudik, and John Langford. Provably efficient rl with rich observations via latent state decoding. In *International Conference on Machine Learning*, pages 1665–1674. PMLR, 2019.

Simon Du, Sham Kakade, Jason Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear classes: A structural framework for provable generalization in rl. In *International Conference on Machine Learning*, pages 2826–2836. PMLR, 2021.

Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. Is a good representation sufficient for sample efficient reinforcement learning? In *International Conference on Learning Representations*, 2020.

Dylan J Foster, Alexander Rakhlin, David Simchi-Levi, and Yunzong Xu. Instance-dependent complexity of contextual bandits and reinforcement learning: A disagreement-based perspective. *arXiv preprint arXiv:2010.03104*, 2020.

Jiafan He, Dongruo Zhou, and Quanquan Gu. Logarithmic regret for reinforcement learning with linear function approximation. In *International Conference on Machine Learning*. PMLR, 2021.

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low bellman rank are pac-learnable. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2017.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR, 2020.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Satinder P Singh Tommi Jaakkola Michael and I Jordan. Reinforcement learning with soft state aggregation. *Advances in neural information processing systems 7*, 7:361, 1995.

Yifei Min, Tianhao Wang, Dongruo Zhou, and Quanquan Gu. Variance-aware off-policy evaluation with linear function approximation. *Advances in neural information processing systems*, 34, 2021.

Dipendra Misra, Mikael Henaff, Akshay Krishnamurthy, and John Langford. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. In *International conference on machine learning*, pages 6961–6971. PMLR, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Aditya Modi, Jinglin Chen, Akshay Krishnamurthy, Nan Jiang, and Alekh Agarwal. Model-free representation learning and exploration in low-rank mdps. *arXiv preprint arXiv:2102.07035*, 2021.

Matteo Papini, Andrea Tirinzoni, Aldo Pacchiano, Marcello Restelli, Alessandro Lazaric, and Matteo Pirotta. Reinforcement learning in linear mdps: Constant regret and representation selection. 2021a.

Matteo Papini, Andrea Tirinzoni, Marcello Restelli, Alessandro Lazaric, and Matteo Pirotta. Leveraging good representations in linear contextual bandits. In *International Conference on Machine Learning*. PMLR, 2021b.

Shuang Qiu, Lingxiao Wang, Chenjia Bai, Zhuoran Yang, and Zhaoran Wang. Contrastive ucb: Provably efficient contrastive self-supervised learning in online reinforcement learning. In *International Conference on Machine Learning*, pages 18168–18210. PMLR, 2022.

Balaraman Ravindran and Andrew G Barto. Model minimization in hierarchical reinforcement learning. In *International Symposium on Abstraction, Reformulation, and Approximation*, pages 196–211. Springer, 2002.

Max Simchowitz and Kevin Jamieson. Non-asymptotic gap-dependent regret bounds for tabular mdps. *Advances in neural information processing systems*, 2019.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. *arXiv preprint arXiv:2009.08319*, 2020.

Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Model-based rl in contextual decision processes: Pac bounds and exponential improvements over model-free approaches. In *Conference on learning theory*, pages 2898–2933. PMLR, 2019.

Masatoshi Uehara, Xuezhou Zhang, and Wen Sun. Representation learning for online and offline rl in low-rank mdps. *arXiv preprint arXiv:2110.04652*, 2021.

Ruosong Wang, Dean Foster, and Sham M Kakade. What are the statistical limits of offline rl with linear function approximation? In *International Conference on Learning Representations*, 2020.

Xinqi Wang, Qiwen Cui, and Simon S Du. On gap-dependent bounds for offline reinforcement learning. *arXiv preprint arXiv:2206.00177*, 2022.

Chen-Yu Wei, Mehdi Jafarnia Jahromi, Haipeng Luo, and Rahul Jain. Learning infinite-horizon average-reward mdps with linear function approximation. In *International Conference on Artificial Intelligence and Statistics*, pages 3007–3015. PMLR, 2021.

Kunhe Yang, Lin Yang, and Simon Du. Q-learning with logarithmic regret. In *International Conference on Artificial Intelligence and Statistics*, pages 1576–1584. PMLR, 2021.

Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International Conference on Machine Learning*, pages 6995–7004. PMLR, 2019.

Lin Yang and Mengdi Wang. Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound. In *International Conference on Machine Learning*, pages 10746–10756. PMLR, 2020.

Mengjiao Yang, Bo Dai, Ofir Nachum, George Tucker, and Dale Schuurmans. Offline policy selection under uncertainty. *arXiv preprint arXiv:2012.06919*, 2020.

Ming Yin, Yaqi Duan, Mengdi Wang, and Yu-Xiang Wang. Near-optimal offline reinforcement learning with linear representation: Leveraging variance information with pessimism. In *International Conference on Learning Representations*, 2022.

Tianjun Zhang, Tongzheng Ren, Mengjiao Yang, Joseph Gonzalez, Dale Schuurmans, and Bo Dai. Making linear mdps practical via contrastive representation learning. In *International Conference on Machine Learning*, pages 26447–26466. PMLR, 2022.

Dongruo Zhou, Jiafan He, and Quanquan Gu. Provably efficient reinforcement learning for discounted mdps with feature mapping. In *International Conference on Machine Learning*. PMLR, 2021.