Q-Learning with Shift-Aware Upper Confidence Bound in Non-Stationary Reinforcement Learning

Ha Manh Bui **Felix Parker** **Kimia Ghobadi** **Anqi Liu**
Johns Hopkins University, Baltimore, MD, U.S.A.
{hbui13, fparker9, kimia, aliu.cs}@jhu.edu

Abstract

We study the Non-Stationary Reinforcement Learning (RL) under distribution shifts in both finite-horizon episodic and infinite-horizon discounted Markov Decision Processes (MDPs). In the finite-horizon case, the transition functions may suddenly change at a particular episode. In the infinite-horizon setting, such changes can occur at an arbitrary time step during the agent's interaction with the environment. While the Q-learning Upper Confidence Bound algorithm (QUCB) can discover a proper policy during learning, due to the distribution shifts, this policy can exploit suboptimal rewards after the shift happens. To address this issue, we propose Density-QUCB (DQUCB), a shift-aware Q-learning UCB algorithm, which uses a transition density function to detect distribution shifts, then leverages its likelihood to enhance the uncertainty estimation quality of Q-learning UCB, resulting in a balance between exploration and exploitation. Theoretically, we prove that our oracle DQUCB achieves a better regret guarantee than QUCB. Empirically, our DQUCB enjoys the computational efficiency of model-free RL and outperforms QUCB baselines by having a lower regret across RL tasks, as well as a COVID-19 patient hospital allocation task using a Deep-Q-learning architecture.

1 INTRODUCTION

Non-stationary Reinforcement Learning (RL) is a sequential decision-making problem in which an agent interacts with a changing environment over time to maximize rewards (Sutton and Barto, 2018). One instance of this setting is considering a run of Markov Decision Process (MDP) dynamics in the finite-horizon episodic MDP (Jin et al., 2018; Menard et al., 2021), where the reward and transition functions can change within an episode. To handle this case, Jin et al. (2018) introduces the Q-learning UCB (i.e., QUCB) algorithm by combining Q-learning (Watkins and Dayan, 1992) with the idea of Upper Confidence Bound (UCB) in bandit literature (Kearns and Singh, 2002; Lattimore and Szepesvári, 2020). On the one hand, Q-learning tries to estimate the optimal Quality(Q)-function, i.e., the state-action value function. With a Q-function, at every state, the agent can greedily select the action with the largest Q-value to interact with the environment. On the other hand, the UCB term (calculated based on the number of visited state-action pairs) measures the uncertainty of the Q-function, aiming to balance the exploration and exploitation performance. Intuitively, when the UCB term is large, the model will be more uncertain. Conversely, when the UCB term is small, the model will be more certain about the Q-values. By leveraging this uncertainty information of UCB, Jin et al. (2018); Wang et al. (2020) show that QUCB is provably efficient in non-stationary RL. Although having a higher regret than model-based RL (e.g., UCBVI, UCBMQ, etc.), QUCB is model-free and typically simpler, enjoys better time and space complexity, and thus is more prevalent in infinite-state environments with modern Deep-RL architectures (Mnih et al., 2015; Wang et al., 2020; Chai et al., 2025).

Method	Shift-Aware ability	Computational efficiency	Handle infinite-state & Deep-RL
QUCB	✗	✓	✓
UCBVI	✓	✗	✗
UCBMQ	✓	✗	✗
Ours	✓	✓	✓

Table 1: A comparison between methods regarding shift-aware ability, computational efficiency (time & space), and ability to handle infinite-state & Deep-RL.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s).

That said, QUCB and the methods mentioned above only consider the finite-horizon episodic MDP setting, where reward and transition functions can change within an episode but are fixed across different episodes. Similarly, in the infinite-horizon discounted MDP, Wang et al. (2020); Yang et al. (2021) assume the transition function is fixed across time steps. This is a strong assumption because the transition function can suddenly change in the real world. Therefore, we study a more challenging and realistic scenario by considering non-stationary RL under distribution shifts, i.e., the transition function can additionally change at a particular episode in the finite-horizon episodic MDP, or at a time step in the infinite-horizon discounted MDP. These settings are critical and align with real-world applications. E.g., in robot navigation, over multiple episodes (interaction trials with the environment), a robot needs to learn an optimal route to the final destination, as shown in Fig. 1. There are slippery sections on the route that may cause the robot to move in an undesirable direction. Over time, the environment, i.e., the slippery levels, can change across episodes due to weather conditions. Because of this distribution shift, the optimal policy can change (e.g., route A with more slippery than B, now becomes less slippery than B). As not designed to handle this kind of shift, QUCB with non-shift-aware ability, may exploit sub-optimal reward by an old policy (learned policy before the shift), resulting in a bad performance after the shift occurs.

To address the non-shift-aware issue in QUCB, we introduce Density-QUCB (DQUCB), a shift-aware Q-learning UCB algorithm, which uses a transition density function to adjust the exploration rate of the UCB term. The key idea of our method is leveraging the likelihood of this density function as an out-of-distribution detector, i.e., when there is no shift, its likelihood will be high, and our model will be more certain (exploitation). In contrast, its likelihood will be low when the distribution shift occurs, which means our model will be more uncertain (exploration). This helps our model improve UCB uncertainty quantification, be able to explore new policies, and avoid exploiting an old policy when the shift happens, resulting in a better regret guarantee than the non-shift-aware QUCB algorithm.

Our contributions are outlined in Tab. 1 and as follows:

- We propose DQUCB, a shift-aware Q-learning UCB algorithm to enhance the uncertainty quantification quality of UCB with Q-learning, resulting in a balance between exploration and exploitation in the non-stationary RL under distribution shifts.
- We theoretically prove the regret of our DQUCB algorithm in the finite-horizon episodic MDP is at most $\mathcal{O}\left(\sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|K\log(|\mathcal{S}||\mathcal{A}|KH/\delta)}\right)$, where $|\mathcal{S}|$, $|\mathcal{A}|$, H, K, ϵ are the number of state, action, planning horizon, episodes, and density estimator error, correspondingly. And in infinite-horizon discounted MDP is at most $\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{(1-\gamma)^6\min\{\Delta_{\min},\bar{\Delta}_{\min}\}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\min\{\Delta_{\min},\bar{\Delta}_{\min}\}}\right)\right)$, where γ is the discount factor, T is the number of cumulative steps, and $\Delta_{\min}, \bar{\Delta}_{\min}$ are the minimum sub-optimality gap. Notably, our oracle DQUCB algorithm (i.e., when $\epsilon \to 0$) is strictly better than the non-shift-aware version (i.e., QUCB) in both finite-horizon episodic and infinite-horizon discounted MDP settings.
- We empirically show DQUCB outperforms QUCB by achieving lower regret on GridWorld and FrozenLake. In addition, it exhibits the computational efficiency of model-free RL and outperforms model-based RL (e.g., UCBVI, UCBMQ) in space and time complexity (see Fig. 4). Furthermore, we extend this framework to neural-network versions on CartPole, COVID-19 patient hospital allocation task, and show it consistently achieves a lower cumulative regret than other Deep Q-learning baselines.

2 PRELIMINARY

In this paper, we consider two RL settings in an unknown MDP, including when the learning agent interacts with the system with (i.e., finite-horizon episodic MDP) and without (i.e., infinite-horizon discounted MDP) any reset. For simplicity, our formal setting considers one time shift. Our results can also be extended to multiple time shifts using our general proof techniques or a similar analysis. We formally define these settings under distribution shifts as follows:

2.1 Finite-horizon episodic MDP under distribution shifts

In the first setting, an episodic Markov Decision Process (MDP) is a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, H, \mathbb{P}, r\rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $H \in \mathbb{N}$ is the planning horizon, $\mathbb{P}_h : \mathcal{S}\times\mathcal{A} \to \Delta(\mathcal{S})$ is the transition operator at step h that takes a state-action pair and returns a distribution over states, and $r_h : \mathcal{S}\times\mathcal{A} \to [0,1]$ is the deterministic reward function at step h. For a finite-horizon episodic MDP, the agent interacts with the MDP for $K \in \mathbb{N}$ episodes. For each episode $k \in [K]$, the agent starts at an initial state $s_1 \in \mathcal{S}$ picked by an adversary (Jin et al., 2018; Menard et al., 2021). In this setting, we focus on the distribution shift of the transition operator $\mathbb{P}_h$ at a particular episode $\bar{K} \in [K]$. Let us denote $\bar{\mathbb{P}}_h$ as the transition operator after the distribution shift occurs. Hence, $\bar{s}_h$ means the state r.v. sampled from $\bar{\mathbb{P}}_h$. Given a policy π which is a sequence of mappings $\pi_h : \mathcal{S} \to \mathcal{A}$ for $h \in [H]$, for a

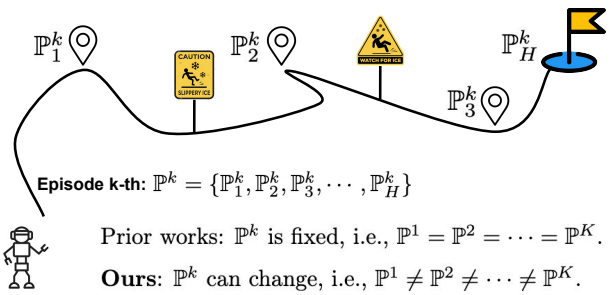


Figure 1: Robot navigation example, where $\mathbb{P}^k$ represents the set of slippery distributions on the route $\{\mathbb{P}_h\}_{h\in[H]}$ at episode k-th. We consider a more general case of prior works. Specifically, prior works set $\{\mathbb{P}_h\}_{h\in[H]}$ stay the same over episode $k \in [K]$, we consider $\{\mathbb{P}_h\}_{h\in[H]}$ can change across different episodes $k \in [K]$. Similarly, in the infinite-horizon discounted MDP, prior works consider a fixed $\mathbb{P}$ across time steps, while we consider that $\mathbb{P}$ can change.

state $s \in \mathcal{S}$, the value function of state $s \in \mathcal{S}$ at the h-step are defined as

$$V_h^\pi(s) := \mathbb{E}\left[\sum_{h'=h}^{H} r_{h'}(s_{h'}, \pi_{h'}(s_{h'}))\Big| s_h = s\right],$$
$$\bar{V}_h^\pi(s) := \mathbb{E}\left[\sum_{h'=h}^{H} r_{h'}(\bar{s}_{h'}, \pi_{h'}(\bar{s}_{h'}))\Big| \bar{s}_h = s\right], \quad (1)$$

and the associated Q-function of a state-action pair $(s,a) \in \mathcal{S}\times\mathcal{A}$ at the h-step are $Q_h^\pi(s,a) := r_h(s,a) + \mathbb{E}\left[\sum_{h'=h+1}^{H} r_{h'}(s_{h'}, \pi_{h'}(s_{h'}))\Big| s_h = s, a_h = a\right]$ and $\bar{Q}_h^\pi(s,a) := r_h(s,a) + \mathbb{E}\left[\sum_{h'=h+1}^{H} r_{h'}(\bar{s}_{h'}, \pi_{h'}(\bar{s}_{h'}))\Big| \bar{s}_h = s, a_h = a\right]$.
We let π^* and $\bar{\pi}^*$ be the optimal policy s.t. $V^{\pi^*}(s) = V^*(s) = \sup_\pi V^\pi(s)$ and $\bar{V}^{\bar{\pi}^*}(s) = \bar{V}^*(s) = \sup_\pi \bar{V}^\pi(s)$. Hence, $Q^*(s,a)$ and $\bar{Q}^*(s,a)$ mean the Q-function under optimal policy π^* and $\bar{\pi}^*$. respectively, $\forall(s,a)$. For each episode $k \in [K]$, the learning agent specifies a policy π^k, plays π^k for H steps and observes trajectory $(s_1, a_1), \cdots, (s_H, a_H)$. The total number of steps is KH, and the total (expected) regret of an execution instance of the agent is then

$$\text{Regret}(K) := \sum_{k=1}^{\bar{K}} (V_1^* - V_1^{\pi_k})(s_1^k) + \sum_{k=\bar{K}+1}^{K} (\bar{V}_1^* - \bar{V}_1^{\pi_k})(s_1^k). \quad (2)$$

It is worth noting that we consider a general case, rather than a special case of the literature on a non-stationary MDP (Jin et al., 2018; Menard et al., 2021). Specifically, prior works consider the case of transition functions $\mathbb{P}_h$ changing across the time horizon $h \in [H]$ within an episode, i.e., $\mathbb{P}_h$ is different over $h \in [H]$ but the set $\{\mathbb{P}_h\}_{h\in[H]}$ stay the same over episode $k \in [K]$. Meanwhile, our method considers that the transition functions additionally change across episodes, i.e., $\mathbb{P}_h$ differs over $h \in [H]$ and the set $\{\mathbb{P}_h\}_{h\in[H]}$ can also change across different episodes $k \in [K]$ (see Fig. 1).

2.2 Infinite-horizon discounted MDP under distribution shifts

In the second setting, we consider a tuple $\mathcal{M}_\gamma := \langle \mathcal{S}, \mathcal{A}, \gamma, \mathbb{P}, r\rangle$, where γ is the discount factor, $\mathbb{P}$ is the transition operator, and r is the reward function (Wang et al., 2020; Yang et al., 2021). We focus on the distribution shift of the transition operator $\mathbb{P}$ at a particular time step $\bar{T} \in [T]$, where $T \in \mathbb{N}$ is the number of first time steps. Let us denote $\bar{\mathbb{P}}$ as the transition operator after the distribution shift occurs. Hence, $\bar{s}$ means the state r.v. sampled from $\bar{\mathbb{P}}$. Let $\mathcal{C} := \{\mathcal{S}\times\mathcal{A}\times[0,1]\}^*\times\mathcal{S}$ be the set of all possible trajectories of any length. A non-stationary deterministic policy $\pi : \mathcal{C} \to \mathcal{A}$ is a mapping from paths to actions. The V function and Q function are defined as follows

$$V^\pi(s) := \mathbb{E}\left[\sum_{i=1}^{\infty} \gamma^{i-1} r(s_i, \pi(c_i))\Big| s_1 = s\right],$$
$$\bar{V}^\pi(s) := \mathbb{E}\left[\sum_{i=1}^{\infty} \gamma^{i-1} r(\bar{s}_i, \pi(\bar{c}_i))\Big| \bar{s}_1 = s\right], \quad (3)$$

and $Q^\pi(s,a) := r(s,a) + \mathbb{E}\left[\sum_{i=2}^{\infty} \gamma^{i-1} r(s_i, \pi(c_i))\Big| s_1 = s, a_1 = a\right]$, $\bar{Q}^\pi(s,a) := r(s,a) + \mathbb{E}\left[\sum_{i=2}^{\infty} \gamma^{i-1} r(\bar{s}_i, \pi(\bar{c}_i))\Big| \bar{s}_1 = s, a_1 = a\right]$, where $(c_i := (s_1, \pi(s_1), r(s_1, \pi(s_1)), \cdots, s_i))$ and $(\bar{c}_i := (\bar{s}_1, \pi(\bar{s}_1), r(\bar{s}_1, \pi(\bar{s}_1)), \cdots, \bar{s}_i))$. Consider the interaction with the environment that starts at state s_1, a learning agent specifies an initial non-stationary policy π_1. At each time step t, the agent takes action $\pi_t(s_t)$, observe r_t and s_{t+1}, and updates π_t to π_{t+1}. The total regret of the agent for the first T steps is thus defined as

$$\text{Regret}(T) := \sum_{t=1}^{\bar{T}} (V^* - V^{\pi_t})(s_t) + \sum_{t=\bar{T}+1}^{T} (\bar{V}^* - \bar{V}^{\pi_t})(s_t). \quad (4)$$

It is worth noting that while Wang et al. (2020) considers a fixed transition function $\mathbb{P}$ across time steps, we consider a more general case by $\mathbb{P}$ can change at a particular time step $t \in [T]$.

3 SHIFT-AWARE DENSITY QUCB

To handle the aforementioned settings, based on Q-learning, we next introduce our main methodology.

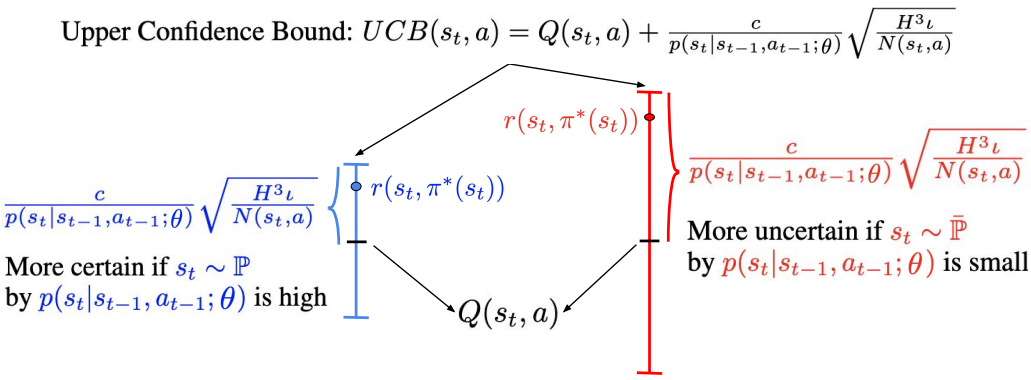


Figure 2: Our shift-aware Q-learning UCB can be more uncertain (i.e., more exploration) if the environment changes because of a low likelihood from $p(\cdot|s, a; \theta)$, and more certain (i.e., more exploitation) if the environment stays the same because of a high likelihood from $p(\cdot|s, a; \theta)$.

In the finite-horizon episodic MDP, our Alg. 1 maintains Q values, $Q_h(s, a)$, for all $(s, a, h) \in \mathcal{S} \times \mathcal{A} \times [H]$ and the corresponding V value, i.e.,

$$V_h(s) \leftarrow \min\{H, \max_{a' \in \mathcal{A}} Q_h(s, a')\}. \tag{5}$$

If at time step $h \in [H]$, the state is $s \in \mathcal{S}$ and the algorithm takes action $a \in \mathcal{A}$ that maximizes the current estimate $Q_h(s, a)$, and is apprised of the next state $s' \in \mathcal{S}$. The algorithm then updates the Q values by

$$\begin{aligned} Q_h(s, a) \leftarrow (1 - \alpha_t) Q_h(s, a) \\ + \alpha_t [r_h(s, a) + V_{h+1}(s') + b_t], \end{aligned} \tag{6}$$

where $t = N_h(s, a)$ is the number of times the algorithm has visited the state-action pair (s, a) at h, α_t is the learning rate, and b_t is the confidence value indicating how certain the algorithm is about the current state-action pair, and are defined as follows

$$\alpha_t := \frac{H+1}{H+t}, \quad b_t := \frac{c}{p(s_{h+1}|s_h, a_h; \theta_h)} \sqrt{H^3 \iota / t}, \tag{7}$$

where $\iota = \log(|\mathcal{S}||\mathcal{A}|KH/\delta)$ and $p(s_{h+1}|s_h, a_h; \theta_h)$ is the likelihood value of density function $p(\cdot|s, a; \theta_h)$ parameterized by parameter θ_h, at state s_{h+1} condition on previous state-action (s_h, a_h), and is computed following Bayes's rule as follows

$$p(s_{h+1}|s_h, a_h; \theta_h) = \frac{p(s_{h+1}, s_h, a_h; \theta_h)}{p(s_h, a_h; \theta_h)}, \tag{8}$$

where parameter θ_h is updated by using maximum likelihood estimation with n most recent tuples (s', s, a) from the interaction with the environment. Specifically about our density function $p(s'|s, a; \theta_h)$ above, we use $p(s'|s, a; \theta_h)$ to estimate the likelihood of sample (s', s, a) from the probability density ratio, i.e., $p(s'|s, a; \theta_h) \approx f_X(s'|s, a)/f_Y(s'|s, a)$, where $(s', s, a) \sim \mathbb{P}_Y$ and $\theta_h = \arg\max_{\theta_h} p(D_n|\theta_h)$, with $D_n \sim \mathbb{P}_X$ is the cumulative dataset with n data points. When before the shift, $(s', s, a) \sim \mathbb{P}_h$ and $D_n \sim \mathbb{P}_h$, thus $\mathbb{P}_X = \mathbb{P}_h$ and $\mathbb{P}_Y = \mathbb{P}_h$. When the shift occurs at episode $\bar{K}$, $(s', s, a) \sim \bar{\mathbb{P}}_h$ and the cumulative dataset $D_n \sim \mathbb{P}_h$, thus $\mathbb{P}_X = \mathbb{P}_h$ and $\mathbb{P}_Y = \bar{\mathbb{P}}_h$. After episode $\bar{K}$, the cumulative dataset accumulates data from the new distribution, i.e., $D_n \sim \bar{\mathbb{P}}_h$, hence for $(s', s, a) \sim \bar{\mathbb{P}}_h$, we have $\mathbb{P}_X = \bar{\mathbb{P}}_h$ and $\mathbb{P}_Y = \bar{\mathbb{P}}_h$.

From the uncertainty term b_t in Eq. 7, we can see that while t is monotonically increasing, $p(s_{h+1}|s_h, a_h; \theta_h)$ is initialized to estimate the density of (s', s, a) sampled from transition operator $\mathbb{P}_h$, over time, the likelihood $p(s_{h+1}|s_h, a_h; \theta_h)$ will be high, yielding the exploration rate in b_t to be low, i.e., the algorithm will be more certain (i.e., exploitation) about its prediction from the mean $Q_h(s, a)$. When the shift happens at episode $\bar{K}$, $p(s_{h+1}|s_h, a_h; \theta_h)$ behaves as an out-of-distriubiton detector because its likelihood will measure the ratio of $\mathbb{P}_h/\bar{\mathbb{P}}_h$, decreasing value, yielding the exploration rate in b_t will be high, i.e., the algorithm will be more uncertain (i.e., exploration) about its prediction from the mean $Q_h(s, a)$. We show this key idea in Fig. 2.

Remark 3.1. **Comparison with model-based RL**. It is worth noticing that although our function $p(\cdot|s, a; \theta_h)$ estimates the density of samples from the transition function, it does not explicitly model the transition operator $\mathbb{P}_h$ like the model-based method (e.g., UCBVI, UCBMQ, etc.), which needs to store and iterate through all possible $(s', s, a) \in \mathcal{S} \times \mathcal{S} \times \mathcal{A}$ tuples with $\tilde{\mathcal{O}}(KH|S|^2|\mathcal{A}|)$ time complexity and $\mathcal{O}(|\mathcal{S}|^2|\mathcal{A}|H)$ space complexity (Auer et al., 2008). Indeed, our training step for θ of $p(\cdot|s, a; \theta_h)$ only depends on the number of cumulative observed (s', s, a) tuples with n window-size. In general, $p(\cdot|s, a; \theta_h)$ can be any density function. In our experiment, we use Kernel Density Estimation (KDE). Therefore, the time complexity of our DQUCB algorithm is $\mathcal{O}(KHn^2)$, and $\mathcal{O}(|\mathcal{S}||\mathcal{A}|Hn)$, which is

Algorithm 1 Our DQUCB algorithm in the finite-horizon episodic MDP

Initialize $Q_h(s,a) \leftarrow H$ & $N_h(s,a) \leftarrow 0$ $\forall(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]$, $p(\cdot;\theta_h)$ $\forall h \in [H]$, $\alpha_t = \frac{H+1}{H+t}$, $\iota = \log(|\mathcal{S}||\mathcal{A}|KH/\delta)$
for every episode $k \in [K]$ **do**
  Receive s_0
  for step $h \in [H]$ **do**
    Take $a_h \leftarrow \arg\max_{a' \in \mathcal{A}} Q_h(s_h, a')$, observe s_{h+1}
    $t = N_h(s_h, a_h) \leftarrow N_h(s_h, a_h) + 1$
    $b_t \leftarrow \frac{c}{p(s_{h+1}|s_h,a_h;\theta_h)}\sqrt{H^3\iota/t}$
    $Q_h(s_h,a_h) \leftarrow (1-\alpha_t)Q_h(s_h,a_h) + \alpha_t\left[r_h(s_h,a_h) + V_{h+1}(s_{h+1}) + b_t\right]$
    $V_h(s_h) \leftarrow \min\{H, \max_{a' \in \mathcal{A}} Q_h(s_h,a')\}$
    Update θ_h of model $p(\cdot;\theta_h)$ by $p(s_{h+1}|s_h,a_h;\theta_h)$
  end for
end for

Algorithm 2 Our DQUCB algorithm in the infinite-horizon discounted MDP

Initialize $Q(s,a) \leftarrow \frac{1}{1-\gamma}$ & $N(s,a) \leftarrow 0$ $\forall(s,a) \in \mathcal{S} \times \mathcal{A}$, $p(\cdot;\theta)$, $\alpha_k = \frac{H+1}{H+k}$, $\iota(k) = \log(|\mathcal{S}||\mathcal{A}|T(k+1)(k+2))$, $H \leftarrow \frac{\ln(2/(1-\gamma))}{\ln(1/\gamma)}$
for every step $t \in [T]$ **do**
  Take $a_t \leftarrow \arg\max_{a' \in \mathcal{A}} Q(s_t, a')$, observe s_{t+1}
  $k = N(s_t,a_t) \leftarrow N(s_t,a_t) + 1$
  $b_k \leftarrow \frac{c_2}{(1-\gamma)p(s_{t+1}|s_t,a_t;\theta)}\sqrt{H\iota(k)/k}$
  $\hat{V}(s_{t+1}) \leftarrow \max_{a' \in \mathcal{A}} \hat{Q}(s_{t+1}, a')$
  $Q(s_t,a_t) \leftarrow (1-\alpha_k)Q(s_t,a_t) + \alpha_k\left[r(s_t,a_t) + \gamma\hat{V}(s_{t+1}) + b_k\right]$
  $\hat{Q}(s_t,a_t) \leftarrow \min\left\{\hat{Q}(s_t,a_t), Q(s_t,a_t)\right\}$
  Update θ of model $p(\cdot;\theta)$ by $p(s_{t+1}|s_t,a_t;\theta)$
end for

much computationally efficient than model-based baselines (e.g., Fig. 4). Note that we can also further improve our algorithm's complexity by applying other density estimation techniques that have near-linear time and space complexity (Chan et al., 2014; Bousquet et al., 2019; Backurs et al., 2019).

In the infinite-horizon discounted MDP, our Alg. 2 maintains Q values, $Q(s,a)$, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and the corresponding $\hat{V}$ value, i.e.,

$$\hat{V}(s_{t+1}) \leftarrow \max_{a' \in \mathcal{A}} \hat{Q}(s_{t+1}, a'),$$
$$\hat{Q}(s_t,a_t) \leftarrow \min\left\{\hat{Q}(s_t,a_t), Q(s_t,a_t)\right\}. \tag{9}$$

If, at time step $t \in [T]$, the state is $s \in \mathcal{S}$, the algorithm takes the action $a \in \mathcal{A}$ that maximizes the current estimate $Q(s,a)$, and is apprised of the next sate $s' \in \mathcal{S}$. The algorithm then updates the Q values as follows

$$Q(s_t,a_t) \leftarrow (1-\alpha_k)Q(s_t,a_t) + \alpha_k\left[r(s_t,a_t) + \gamma\hat{V}(s_{t+1}) + b_k\right], \tag{10}$$

where $k = N(s,a)$ is the number of times the algorithm has visited the state-action pair (s,a), α_k is the learning rate, and b_k is the confidence value indicating how certain the algorithm is about the current state-action pair, and are defined as follows

$$\alpha_k := \frac{H+1}{H+k}, b_k := \frac{c_2}{(1-\gamma)p(s_{t+1}|s_t,a_t;\theta)}\sqrt{H\iota(k)/k}, \tag{11}$$

where $H \leftarrow \frac{\ln(2/(1-\gamma))}{\ln(1/\gamma)}$, $\iota(k) = \log(|\mathcal{S}||\mathcal{A}|T(k+1)(k+2))$, and $p(s_{t+1}|s_t,a_t;\theta)$ is the likelihood value of density function $p(\cdot|s,a;\theta)$ at state s_{t+1} condition on previous state-action (s_t,a_t), and is computed following Bayes's rule as follows

$$p(s_{t+1}|s_t,a_t;\theta) = \frac{p(s_{t+1},s_t,a_t;\theta)}{p(s_t,a_t;\theta)}, \tag{12}$$

where parameter θ is updated by n most recent tuples (s',s,a). From the uncertainty term b_k in Eq. 11, we can see that while k is monotonically increasing, $p(s_{t+1}|s_t,a_t;\theta)$ is initialized to estimate the density of (s',s,a) sampled from transition operator $\mathbb{P}$, over time, the likelihood $p(s_{t+1}|s_t,a_t;\theta)$ will be high, yielding the exploration rate in b_k will be low, i.e., the algorithm will be more certain (i.e., exploitation). When the shift happens at time step $\bar{T}$, $p(s_{t+1}|s_t,a_t;\theta)$ behaves as a out-of-distriubiton detector by its likelihood will measure the ratio of $\mathbb{P}/\bar{\mathbb{P}}$, decreasing value, yielding the exploration rate in b_k will be high, i.e., the algorithm will be more uncertain (i.e., exploration).

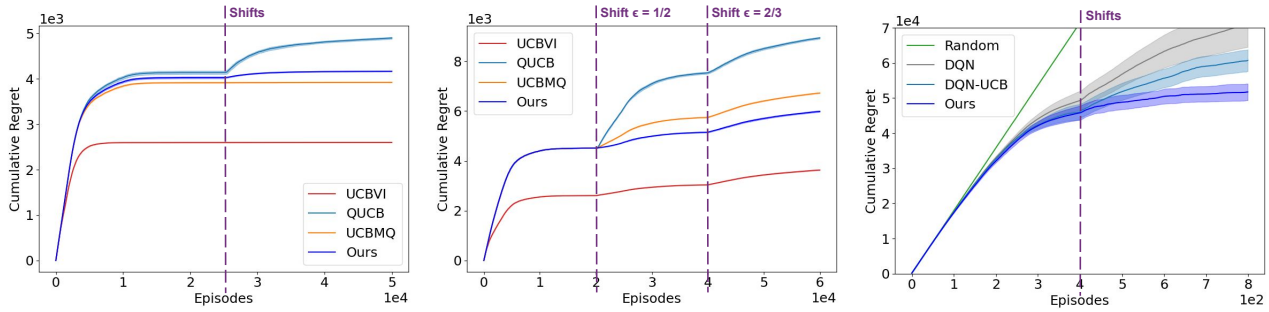
4 THEORETICAL ANALYSIS

To formally explain why our shift-aware DQUCB can improve the uncertainty estimation quality of UCB with Q-learning, resulting in a balance between exploration and exploitation under distribution shifts, we next analyze the cumulative regret of Alg. 1 and Alg. 2. We first provide the regret bound for the Alg. 1 in the finite-horizon episodic MDP under distribution shifts:

Theorem 4.1. *There exists an absolute constant $c > 0$ such that for any $\delta \in (0,1)$, if we choose $b_t = \frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{H^3\iota/t}$, where $t = N_h(s,a)$, then with probability $1-\delta$, the regret of the Algorithm 1 satisfies*

$$\textit{Regret}(K) \leq \mathcal{O}\left(\sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|K\log(|\mathcal{S}||\mathcal{A}|KH/\delta)}\right),$$

where ϵ is the estimator error of $1/p(\cdot|s,a;\theta_h)$.



(a) (b) (c)

Figure 3: (a) Cumulative regret on GridWorld, transition noise $\epsilon = 0.01$ and $\epsilon = 0.2$ before and after the shift at $\bar{K} = 25000$; (b) Cumulative regret on Frozen-Lake, slippery level $\epsilon = 0$, $\epsilon = 1/2$, and $\epsilon = 2/3$ before and after the shift at $\bar{K} = \{20000, 40000\}$; (c) Cumulative regret on CartPole, the transition noise $\mathcal{N}(0, 0.15)$ is added to the velocity state after episode $\bar{K} = 400$. Results are average over 10 runs. We refer to the computational complexity in Fig. 4.

Remark 4.2. Thm. 4.1 shows that when the estimation error $\epsilon \to 0$, our DQUCB in Alg. 1 achieves $\mathcal{O}\left(\sqrt{H^5|\mathcal{S}||\mathcal{A}|K\log(|\mathcal{S}||\mathcal{A}|KH/\delta)}\right)$ regret bound. Meanwhile, due to the distribution shift, the non-shift-aware version, i.e., Q-learning Hoeffding UCB (Jin et al., 2018) with the selection of $b_k = c\sqrt{H^3\iota/t}$, can leads to $\mathcal{O}\left(\sqrt{H^5|\mathcal{S}||\mathcal{A}|\bar{K}\log(|\mathcal{S}||\mathcal{A}|\bar{K}H/\delta)} + H(K-\bar{K})\right)$. Since $\lim_{K\to\infty} \frac{\sqrt{H^5|\mathcal{S}||\mathcal{A}|K\log(|\mathcal{S}||\mathcal{A}|KH/\delta)}}{\sqrt{H^5|\mathcal{S}||\mathcal{A}|\bar{K}\log(|\mathcal{S}||\mathcal{A}|\bar{K}H/\delta)}+H(K-\bar{K})} = 0$ by L'Hôpital's rule, we can see that the regret bound of our shift-aware DQUCB in Alg. 1 when $\epsilon \to 0$, is strictly better than the non-shift-aware version QUCB.

It is worth noticing that in the finite-horizon episodic MDP, the regret in Thm. 4.1 only needs to be measured at the initial state s_1, i.e., $(V_1^* - V_1^\pi)(s_1)$ and $(\bar{V}_1^* - \bar{V}_1^\pi)(s_1)$ in the first KH steps. However, this analysis is non-trivial to extend to the infinite horizon setting because the agent may enter under-explored regions at any time $t \in [T]$ (Wang et al., 2020). Therefore, it is necessary to introduce the sub-optimality gap (Yang et al., 2021), i.e., $\Delta(s,a) := V^*(s) - Q^*(s,a)$ and $\bar{\Delta}(s,a) := \bar{V}^*(s) - \bar{Q}^*(s,a)$ to bound the gap at timestep t and state s_t, i.e., $(V^* - V^{\pi_t})(s_t)$ and $(\bar{V}^* - \bar{V}^{\pi_t})(s_t)$. From this definition, we finally provide the regret bound for Alg. 2 in the infinite-horizon discounted MDP under distribution shifts:

Theorem 4.3. *There exists an absolute constant $c_2 > 0$ such that for any $\delta \in (0,1)$, if we choose $b_k = \frac{c_2}{p(\cdot|s,a;\theta)}\sqrt{H\iota(k)/k}$, then with probability $1-\delta$, the regret of the Algorithm 2 satisfies*

$$Regret(T) \leq \mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{(1-\gamma)^6 \min\{\Delta_{\min}, \bar{\Delta}_{\min}\}} \cdot \log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\min\{\Delta_{\min}, \bar{\Delta}_{\min}\}}\right)\right),$$

where ϵ is the estimator error of $1/p(\cdot|s,a;\theta)$, $\Delta_{\min} := \min_{(s,a)\in\mathcal{S}\times\mathcal{A}}\{\Delta(s,a) : \Delta(s,a) \neq 0\}$, and $\bar{\Delta}_{\min} := \min_{(s,a)\in\mathcal{S}\times\mathcal{A}}\{\bar{\Delta}(s,a) : \bar{\Delta}(s,a) \neq 0\}$.

The proof for Thm. 4.1 and Thm. 4.3 decompose the regret to bounding the learning error before and after the shift at episode $\bar{K}$ and time step $\bar{T}$, use the density function to construct the concentration bound of Q-value with Lem. A.2 and Lem. A.3, and finally adapt techniques of the learning error recursion from (Jin et al., 2018; Wang et al., 2020), details are in Apd. A.3 and Apd. A.4, respectively.

Remark 4.4. Thm. 4.3 shows that when the estimation error $\epsilon \to 0$, our DQUCB in Alg. 2 achieves $\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|}{(1-\gamma)^6\min\{\Delta_{\min},\bar{\Delta}_{\min}\}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\min\{\Delta_{\min},\bar{\Delta}_{\min}\}}\right)\right)$ regret bound. Meanwhile, due to the distribution shift, the non-shift-aware version, i.e., Q-learning Hoeffding UCB (Wang et al., 2020) with the selection of $b_k = c_2\sqrt{H\iota(k)/k}$, can leads to $\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|}{(1-\gamma)^6\Delta_{\min}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|\bar{T}}{(1-\gamma)\Delta_{\min}}\right)\right) + (T-\bar{T})$. Similar to the limit of a function analysis in Remark 4.2, when $T \to \infty$, we can see that the regret bound of our shift-aware DQUCB in Alg. 2 is strictly better than the non-shift-aware version QUCB.

Justification on how density estimator error ϵ scales with time or sample. The density estimator error ϵ depends on the specific density estimation methods. In our experiment, we use the Kernel Density Estimation, hence, the $\epsilon = \mathcal{O}(n_t^{-4/5})$, where n_t is the number of cumulative data points at time t. For other parametric methods, the error can be reduced to $\epsilon = \mathcal{O}(n_t^{-1})$ (Wahba, 1975). In general, over time, the more samples we have, the smaller the estimator error bound, leading to a smaller gap between our oracle and practical algorithm.

5 EXPERIMENTS

5.1 Experimental settings

We illustrate the benefits of our DQUCB across four different tasks. In the first two tasks, we compare with other tabular UCB-based RL baselines (i.e., QUCB (Jin et al., 2018), UCBMQ (Menard et al., 2021), and UCVI (Azar et al., 2017)) in the GridWorld and Frozen Lake environments. In the third task, we extend our DQUCB to an infinite-state with Deep-RL on CartPole-v0 (Brockman et al., 2016). In the final task, we extend this to a real-world healthcare application. We optimize KDE with the most recent $n = 100$ cumulative samples. In the Deep-RL setting, since the UCB term needs to store the number of visited pairs $N_h(s, a)$, we adapt the hashing technique of Tang et al. (2017) to count this number. In GridWorld and Frozen Lake, the shift is the change of the slippery level when moving between states. In CartPole, the shift is the change of the Gaussian noise level to the cart and pole angular velocity features. In the healthcare dataset, the shift is the change of the parameter for updating the estimated COVID-19 occupancy. More details are in Apd. B.1.

5.2 Main results

5.2.1 GridWorld and Frozen Lake

From Fig. 3 (a) and Fig. 3 (b), we observe that our shift-aware DQUCB (colored by dark blue) outperforms the non-shift-aware version (i.e., QUCB) by having a lower cumulative regret across episodes. Notably, we can see that due to being able to detect the distribution shift, our method can explore new optimal policies, resulting in significantly lower regret than QUCB after the shift happens. This result confirms our theoretical guarantee in Sec.4.

It is also worth noticing that while having a higher regret than the model-based RL (i.e., UCVI) and competitive results with the mixture of model-based and model-free RL (i.e., UCBMQ), **our method is much computationally efficient with a remarkably lower time and space complexity in Fig. 4**. This is because UCVI and UCBMQ need to build value functions for each state-action pair, and need to iterate through all possible $(s', s, a) \in \mathcal{S} \times \mathcal{S} \times \mathcal{A}$ tuples per each time step (see Remark 3.1). In contrast, our method only needs to optimize the density function with cumulative observed tuples. Therefore, these results suggest that our method can balance between regret and computational efficiency, and can be deployed in complex environments or low-resource applications in the real world. More results and analysis are provided in Apd. B.3.

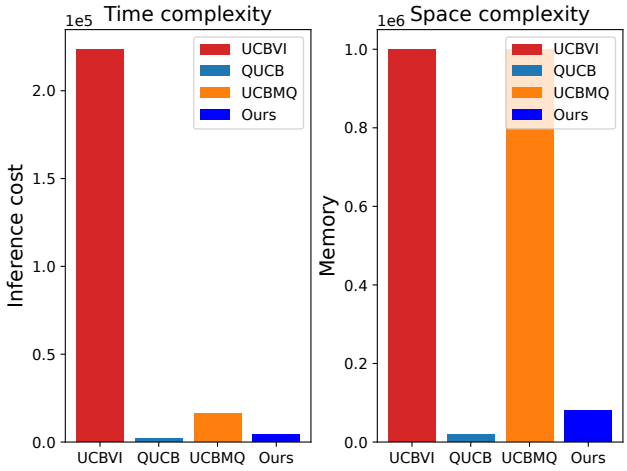


Figure 4: Time and space complexity comparison (lower are better) in the [10] × [5] GridWorld environment.

5.2.2 CartPole

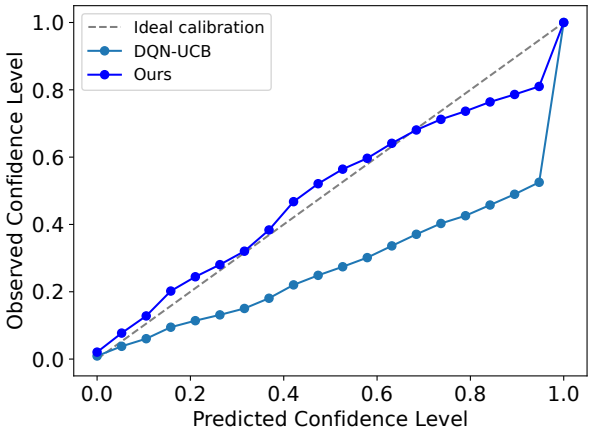
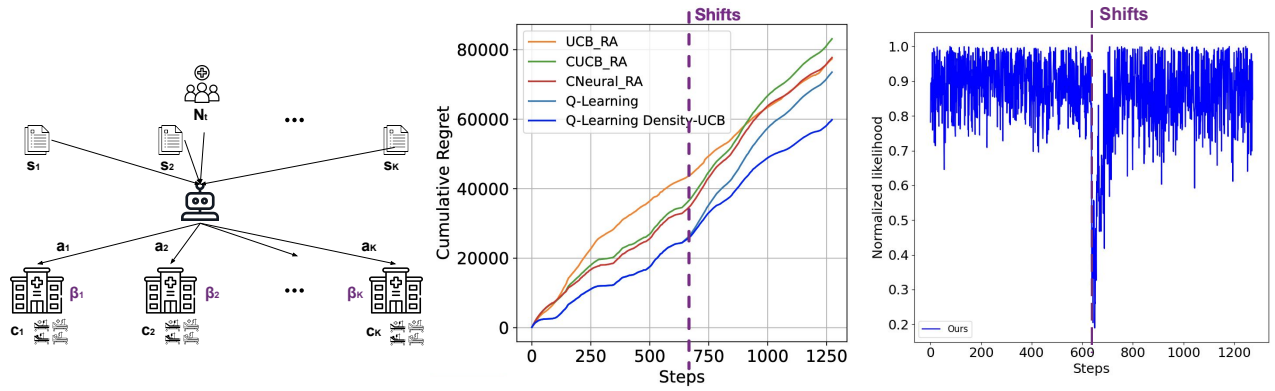


Figure 5: Uncertainty quantification quality of UCB with calibration error measurement.

To illustrate the merits of our method in complex domains, we deploy it in an infinite-state environment with CartPole and Deep-RL. Since the state space is infinite, UCVI and UCBMQ are non-trivial in extending to this setting because they need to iterate through all possible state-action pairs. Hence, we compare our method with 3 baselines, including random policy, Deep Q-Network (DQN) (Paszke et al., 2019), and its UCB extension, i.e., DQN-UCB. Although training Q-learning with neural networks is quite unstable, Fig. 3 (c) still shows that our method achieves the lowest cumulative regret, especially under the distribution shift. This once again confirms our tighter regret bound in Sec. 4. To better understand the UCB quality



(a) (b) (c)

Figure 6: Infinite-horizon discounted MDP setting: (a) Agent receives K hospital status states, needs to allocate N_t patients to K hospitals s.t. minimize the number of overflow; (b) Performance across $K = 40$ Texas hospitals daily from 2020-08 to 2024-01; (c) Normalized likelihood value of our density function across time steps.

with Q-values, Fig. 5 evaluates the calibration performance. Intuitively, calibration means a p confidence interval contains the reward p of the time (Bui et al., 2025). We can see that by being unable to detect the distribution shift, DQN-UCB is over-confident in its prediction. Meanwhile, by leveraging the density function to detect the shift, our method is well-calibrated and significantly better than DQN-UCB.

5.3 COVID-19 patient hospital allocation

Finally, we evaluate our model's performance on a COVID-19 patient hospital allocation. Specifically, given K hospitals in Fig. 6 (a), the agent receives the state $s_t \in \mathbb{R}^{2K+1}$ of a hospital at time t. The state $s_t = (y_{1,t}, \cdots, y_{K,t}, d_{1,t}, \cdots, d_{K,t}, N_t)$, where $y_{i,t}$ and $d_{i,t}$ are the number of COVID-19 patient occupancy and non-COVID patients in hospital i on day t, and N_t is the number of arriving COVID-19 patients in the system on day t. For every day t, the agent must allocate these N_t patients to K hospitals by action $a_t = (a_{1,t}, \cdots, a_{K,t})$ to minimize the number of overflows across hospitals, i.e., maximize the reward $r_t = \sum_{i=1}^{K} -\max(0, (\beta_i \cdot y_{i,t} + a_{i,t+1} + d_{i,t+1}) - c_i)$, where c_i is the capacity of the hospital i and β_i is the parameter provided by the environment to update the estimated occupancy of COVID-19 (details are in Apd. B.1). We use the environment parameter β and the number of arriving COVID-19 patients N_t from the real-world dataset: COVID-19 Reported Patient Impact & Hospital Capacity by Facility, provided by the U.S. Department of Health & Human Services over $T = 1274$ days from 2020 to 2024. The environment is shifted by changing β to be uniform at day $\bar{T} = 637$. We use a neural-net to estimate the Q_t matrix, where $Q[i, j]$ means how many people can be saved if we allocate j patients to hospital i. Then, allocate N_t patients to the hospital with action a_t by using an oracle from Q-values (Zuo and Joe-Wong, 2021; Li et al., 2024).

Fig. 6 (b) summarizes our results with the cumulative regret (optimal rewards are obtained from the dataset). We observe that before the shifts happen, our method and DQN-UCB outperform other baselines (Zuo and Joe-Wong, 2021) with the lowest cumulative regret. Notably, when the shift occurs, our density function decreases the likelihood value in Fig. 6 (c), signaling to the model that the environment is changing. This yields our UCB value increase, i.e., the model increases its uncertainty and encourages exploring new policies. As a result, compared to the non-shift-aware DQN-UCB method, our method achieves better adaptation with significantly lower regret after the shift occurs. This result suggests that our method can allocate COVID-19 patients to hospitals better than other baselines, so that we can save more people in the real world.

6 RELATED WORK

UCB-based methods in non-stationary RL. Relying on the principle of optimism in the face of uncertainty, UCB has become an efficient strategy to explore the uncertain environment while maximizing the reward in bandit (Abbasi-yadkori et al., 2011). Building on this direction, a recent line of research has applied these UCB ideas to address non-stationary RL, encompassing both model-based and model-free settings. Specifically, in the model-based RL, UCLR2 (Auer et al., 2008) and UCLR3 (Bourel et al., 2020) form estimates of the transition probabilities of the MDP using past samples, and add UCB to the estimated transition matrix. An alternative approach is UCBVI (Azar et al., 2017), which directly adds a UCB bonus to the Q-values with a strong assumption that transition

matrices are similar within an episode. Despite the guarantees of sharp regret and the match with the lower bound (that is, $\Omega(\sqrt{H^3|\mathcal{S}||\mathcal{A}|K})$), all of the results in this model-based research require estimating and storing the entire transition matrix and thus suffer from unfavorable time (i.e., $\tilde{\mathcal{O}}(KH|\mathcal{S}|^2|\mathcal{A}|)$) and space complexities (i.e., $\mathcal{O}(|\mathcal{S}|^2|\mathcal{A}|H)$).

Therefore, toward a computationally efficient method, Jin et al. (2018) introduces model-free Q-learning UCB by incorporating an UCB term to show the confidence of Q-values, which can achieve total regret $\mathcal{O}(\sqrt{H^5|\mathcal{S}||\mathcal{A}|K})$ with $\mathcal{O}(KH)$ time and $\mathcal{O}(|\mathcal{S}||\mathcal{A}|H)$ space complexity in episodic MDP. Based on this direction, Wang et al. (2020) and Yang et al. (2021) later show that Q-learning UCB achieves nearly optimal regret bound in discounted MDP. In particular, if there exists a strictly positive sub-optimality gap, Wang et al. (2020) proves that Q-learning enjoys a $\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|}{(1-\gamma)^6\Delta_{\min}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\Delta_{\min}}\right)\right)$ cumulative regret bound. Yet, all of these results for Q-learning UCB assume that the transition between episodes in episodic MDP and the step in discounted MDP are fixed. This is a strong assumption in real-world sequential decision-making applications because the environment can change in the long run. Hence, we relax this fixed assumption by considering that the transition function can change at a particular episode and time step.

There are also other variants of non-stationary RL settings, such as broader literature on continual or concept-drift RL (Cheung et al., 2020; Xie et al., 2021). Some close to our setting, like Gajane et al. (2018), present a non-UCB approach that considers a switching-MDP problem, rather than our non-stationary problem. Mao et al. (2021) constrains the transition cumulative variations to not exceed certain variation budgets, and the regret bound depends on this budget value, whereas our setting and analysis do not depend on any budget.

Improving UCB quality. Using a density function to improve model uncertainty has been studied in classification and regression (Bui and Liu, 2024; Manh Bui and Liu, 2024), but only in supervised learning. In sequential decision problems, improving the UCB quality to obtain better regret has shown promising results in bandit (Auer et al., 2002; Kuleshov and Precup, 2014; Zhou et al., 2021). For example, Malik et al. (2019); Deshpande et al. (2024) have empirically shown that calibrated UCB algorithms can result in lower cumulative regret. Theoretically, (Zhao et al., 2023; Bui et al., 2025) show that variance-aware UCB, i.e., using the reward noise variance to improve UCB quality, can further achieve a tighter regret bound. Regarding RL, Strehl et al. (2006) introduced delayed Q-learning, where the Q-value for each state-action pair is updated only once every certain number of times this pair is visited. Yet, it is quite sample-inefficient compared to other approaches (Jin et al., 2018). Closest to our work is UCBMQ (Menard et al., 2021; Bowen et al., 2021; Ghavamzadeh et al., 2011), which is also based on the Q-learning UCB, but additionally incorporates a momentum term that is built from the value functions for each state-action pair. Because of needing to compute these bias-value functions for every round, UCBMQ is significantly less computationally efficient than model-free baselines by its $\mathcal{O}(H(|\mathcal{S}|+|\mathcal{A}|)K)$ time and $\mathcal{O}(|\mathcal{S}|^2|\mathcal{A}|H)$ space complexity. Beyond this computational limitation, UCBMQ is also designed only for tabular episodic MDPs. Meanwhile, our DQUCB is more computationally efficient, designed for both episodic and discounted MDPs, and can extend to Deep-RL with a continuous state-action space.

7 CONCLUSION AND DISCUSSION

Q-learning with UCB exploration has become a standard model-free RL that is provably efficient, with nearly optimal regret bound in non-stationary RL. Yet, it can exploit sub-optimal reward if the environment suddenly changes in some episode in the finite-horizon episodic MDP, or step in the infinite-horizon discounted MDP. To tackle this challenge, we introduce DQUCB, a shift-aware Q-learning UCB that leverages the transition density function to enhance the uncertainty quantification of the UCB, resulting in better exploration and exploitation. Our theoretical results show that our oracle DQUCB can achieve a nearly optimal regret bound in both episodic MDP and discounted MDP under distribution shifts. Our empirical results demonstrate the computational efficiency of our method when compared to model-based baselines and confirm our theoretical analysis with a significantly lower regret than Q-learning baselines across different tasks, datasets, and model architectures. With these results, we hope our work will contribute to the literature on improving the uncertainty and robustness of UCB for Q-learning in sequential problems.

Our proposed method is based on Q-learning, which is specifically designed for discrete action spaces, and requires conversion between continuous and discrete spaces for applications in continuous spaces. In addition, the quality of our UCB relies on the quality of the density function; if the density estimator is not sufficiently accurate, it can affect the performance of our model. Future work includes tackling the limitation of density estimation with the kernel estimator and extending our method to more RL tasks.

Acknowledgement

This work is supported by a seed grant from JHU Institute of Assured Autonomy.

References

Yasin Abbasi-yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, 2011.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 2002.

Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. In *Advances in Neural Information Processing Systems*, 2008.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Arturs Backurs, Piotr Indyk, and Tal Wagner. Space and time efficient kernel density estimation in high dimensions. In *Advances in Neural Information Processing Systems*, 2019.

Hippolyte Bourel, Odalric Maillard, and Mohammad Sadegh Talebi. Tightening exploration in upper confidence reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Olivier Bousquet, Daniel Kane, and Shay Moran. The optimal approximation factor in density estimation. In *Proceedings of the Thirty-Second Conference on Learning Theory*, 2019.

Weng Bowen, Xiong Huaqing, Zhao Lin, Liang Yingbin, and Zhang Wei. Finite-time theory for momentum q-learning. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, 2021.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Ha Manh Bui and Anqi Liu. Density-softmax: Efficient test-time model for uncertainty estimation and robustness under distribution shifts. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Ha Manh Bui, Enrique Mallada, and Anqi Liu. Variance-aware linear UCB with deep representation for neural contextual bandits. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025.

Jiajun Chai, Sicheng Li, Yuqian Fu, Dongbin Zhao, and Yuanheng Zhu. Empowering LLM agents with zero-shot optimal decision-making through q-learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Siu-On Chan, Ilias Diakonikolas, Rocco A. Servedio, and Xiaorui Sun. Near-optimal density estimation in near-linear time using variable-width histograms. In *Advances in Neural Information Processing Systems*, 2014.

Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu. Reinforcement learning for non-stationary Markov decision processes: The blessing of (More) optimism. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Shachi Deshpande, Charles Marx, and Volodymyr Kuleshov. Online calibrated and conformal prediction improves Bayesian optimization. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2024.

Pratik Gajane, Ronald Ortner, and Peter Auer. A sliding-window algorithm for markov decision processes with arbitrarily changing rewards and transitions, 2018.

Mohammad Ghavamzadeh, Hilbert Kappen, Mohammad Azar, and Rémi Munos. Speedy q-learning. In *Advances in Neural Information Processing Systems*, 2011.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, 2018.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 2002.

Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems, 2014.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.

Yikuan Li, Chengsheng Mao, Kaixuan Huang, Hanyin Wang, Zheng Yu, Mengdi Wang, and Yuan Luo. Deep reinforcement learning for efficient and fair allocation of health care resources, 2024.

Ali Malik, Volodymyr Kuleshov, Jiaming Song, Danny Nemer, Harlan Seymour, and Stefano Ermon. Calibrated model-based deep reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Ha Manh Bui and Anqi Liu. Density-regression: Efficient and distance-aware deep regressor for uncertainty estimation under distribution shifts. In *Pro-*

ceedings of The 27th International Conference on Artificial Intelligence and Statistics, 2024.

Weichao Mao, Kaiqing Zhang, Ruihao Zhu, David Simchi-Levi, and Tamer Basar. Near-optimal model-free reinforcement learning in non-stationary episodic mdps. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

Pierre Menard, Omar Darwiche Domingues, Xuedong Shang, and Michal Valko. Ucb momentum q-learning: Correcting the bias without forgetting. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Alexander L. Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L. Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Grace Wahba. Optimal Convergence Properties of Variable Knot, Kernel, and Orthogonal Series Methods for Density Estimation. *The Annals of Statistics*, 1975.

Yuanhao Wang, Kefan Dong, Xiaoyu Chen, and Liwei Wang. Q-learning with ucb exploration is sample efficient for infinite-horizon mdp. In *International Conference on Learning Representations*, 2020.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 1992.

Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst continual structured non-stationarity. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

Kunhe Yang, Lin Yang, and Simon Du. Q-learning with logarithmic regret. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, 2021.

Heyang Zhao, Jiafan He, Dongruo Zhou, Tong Zhang, and Quanquan Gu. Variance-dependent regret bounds for linear bandits and reinforcement learning: Adaptivity and computational efficiency. In *Proceedings of Thirty Sixth Conference on Learning Theory*, 2023.

Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Proceedings of Thirty Fourth Conference on Learning Theory*, 2021.

Jinhang Zuo and Carlee Joe-Wong. Combinatorial multi-armed bandits for resource allocation, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Q-Learning with Shift-Aware Upper Confidence Bound in Non-Stationary Reinforcement Learning (Supplementary Material)

In this supplementary material, we collect proofs and remaining materials deferred from the main paper. In Appendix A, we provide the proofs for all our theoretical results, including: proof of Theorem 4.1 in Appendix A.3; proof of Theorem 4.3 in Appendix A.4; proof of Lemma A.2 in Appendix A.5; proof of Lemma A.3 in Appendix A.6. In Appendix B, we provide additional information about our experiments, including: sufficient details about experimental settings in Appendix B.1; demo code in Appendix B.2; additional results in Appendix B.3 with performance across different shift intensities, types of environment shift, further ablation studies for comparison with model-based RL, density estimation, and extension to continuous action space experiments. Finally, the source code to reproduce our results is available at https://github.com/Angie-Lab-JHU/density-Qlearning-UCB.

A Proofs

In this section, we first formally introduce notations that will be used intensively in our proofs in Appendix A.1, and then summarize our useful lemmas to support our proofs in Appendix A.2.

A.1 Notations

Notation in the finite-horizon episodic MDP. Let $\mathbb{I}[A]$ as the indicator function for event A, (s_h^k, a_h^k) as the actual state-action pair observed and chosen at step h of episode k, and Q_h^k, V_h^k, N_h^k as the Q_h, V_h, N_h functions at the beginning of episode k, respectively. Using this notation and let $t = N_h^k(s,a)$ and suppose (s,a) was previously taken at step h of episodes $k_1, \cdots, k_t < k$, the update equation at episode k in Algorithm 1 can be rewritten as follows, for every $h \in [H]$,

$$Q_h^k(s,a) = \alpha_t^0 H + \sum_{i=1}^{t} \alpha_t^i \left[r_h(s,a) + V_{h+1}^{k_i}(s_{h+1}^{k_i}) + b_i \right], \tag{13}$$

$$V_h^k(s) \leftarrow \min\{H, \max_{a' \in \mathcal{A}} Q_h^k(s,a')\}, \forall s \in \mathcal{S}. \tag{14}$$

Let $[\mathbb{P}_h V_{h+1}](s,a) := \mathbb{E}_{s' \sim \mathbb{P}_h(\cdot|s,a)} V_{h+1}(s')$ and its empirical counterpart of episode k as $\left[\hat{\mathbb{P}}_h V_{h+1}\right](s,a) := V_{h+1}(s_{h+1})$, which is defined only for $(s,a) = (s_h^k, a_h^k)$.

Notation in the infinite-horizon discounted MDP. Let Q^t, $\hat{Q}^t$, V^t, $\hat{V}^t$, N^t are the value of Q, $\hat{Q}$, V, $\hat{V}$, N right before the t-th step, respectively. Let $\tau(s,a,i) = \max\{t : N^t(s,a) = i-1\}$ be the step t at which $(s^t, a^t) = (s,a)$ for the i-th time. From the update rule in the Algorithm 2, we have

$$\hat{Q}^t(s,a) = \alpha_t^0 \frac{1}{1-\gamma} + \sum_{i=1}^{t} \alpha_t^i \left[r(s,a) + \gamma \hat{V}_{t+i}(s_{t_i+1}) + b_t \right], \tag{15}$$

$$Q^t(s,a) = \min\{\hat{Q}^t(s,a), Q^t(s,a)\}, \quad \hat{V}^t(s) \leftarrow \max_{a' \in \mathcal{A}} \hat{Q}^t(s,a'), \forall s \in \mathcal{S}. \tag{16}$$

A.2 Useful Lemmas

Our proofs in this section are based on the following provable lemmas:

Lemma A.1. *(Learning rate properties (Jin et al., 2018; Wang et al., 2020)). For the learning rate $\alpha_t = \frac{H+1}{H+t}$, let $\alpha_t^0 = \prod_{j=1}^t (1-\alpha_j)$, $\alpha_t^i = \alpha_i \prod_{j=i+1}^t (1-\alpha_j)$, then we have $\sum_{i=1}^t \alpha_t^i = 1$ and $\alpha_t^0 = 0$ for $t \geq 1$; $\sum_{i=1}^t \alpha_t^i = 0$ and $\alpha_t^0 = 1$ for $t = 0$. And, the following properties hold for α_t^i:*

(a) $\frac{1}{\sqrt{t}} \le \sum_{i=1}^t \frac{\alpha_t^i}{\sqrt{i}} \le \frac{2}{\sqrt{t}}$ *for every* $t \ge 1$.
(b) $\max_{i\in[t]} \alpha_t^i \le \frac{2H}{t}$ *and* $\sum_{i=1}^t (\alpha_t^i)^2 \le \frac{2H}{t}$ *for every* $t \ge 1$.
(c) $\sum_{t=i}^\infty \alpha_t^i = 1 + \frac{1}{H}$ *for every* $i \ge 1$.
(d) $\sqrt{\frac{\iota(t)}{t}} \le \sum_{i=1}^t \alpha_t^i \sqrt{\frac{\iota(i)}{i}} \le 2\sqrt{\frac{\iota(t)}{t}}$, *where* $\iota(t) = \ln(c(t+1)(t+2))$, *for every* $t \ge 1$, $c \ge 1$.

Lemma A.2. *(Bound on $Q_h^k - Q^*$ and $Q_h^k - \bar{Q}^*$). There exists an absolute constant $c > 0$ s.t., for any $p \in (0,1)$, letting $b_t = \frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{H^3\iota/t}$ with $t = N_h(s,a)$, we have $\beta_t = 2\sum_{i=1}^t \alpha_t^i b_i \le 4\frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{H^3\iota/t}$ and, for simultaneously $\forall (s,a,h,k) \in \mathcal{S}\times\mathcal{A}\times[H]\times[K]$, with probability at least $1-\delta$, it holds that*

$$0 \le (Q_h^k - Q_h^*)(s,a) \le \alpha_t^0 H + \sum_{i=1}^t \alpha_t^i \left(V_{h+1}^{k_i} - V_{h+1}^*\right)(s_{h+1}^{k_i}) + \beta_t,$$

and

$$0 \le (Q_h^k - \bar{Q}_h^*)(s,a) \le \alpha_t^0 H + \sum_{i=1}^t \alpha_t^i \left(V_{h+1}^{k_i} - \bar{V}_{h+1}^*\right)(s_{h+1}^{k_i}) + \beta_t.$$

The proof is in Appendix A.5.

Lemma A.3. *(Bound on $Q^t - Q^*$ and $Q^t - \bar{Q}^*$). There exists an absolute constant $c_2 > 0$ s.t., for any $p \in (0,1)$, letting $t = N_p(s,a)$, $t_i = \tau(s,a,i)$, $b_t = \frac{c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{H\iota(t)/t}$, we have $\beta_t \le \frac{c_3}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{H\iota(t)/t}$ and, for simultaneously $\forall (s,a,t) \in \mathcal{S}\times\mathcal{A}\times[T]$, with probability at least $1-\delta$, it holds that*

$$0 \le (\hat{Q}^t - Q^*)(s,a) \le (Q^t - Q^*)(s,a) \le \frac{\alpha_t^0}{1-\gamma} + \sum_{i=1}^t \gamma\alpha_t^i \left(\hat{V}_{t_i} - V^*\right)(s_{t_i+1}) + \beta_t,$$

and

$$0 \le (\hat{Q}^t - \bar{Q}^*)(s,a) \le (Q^t - \bar{Q}^*)(s,a) \le \frac{\alpha_t^0}{1-\gamma} + \sum_{i=1}^t \gamma\alpha_t^i \left(\hat{V}_{t_i} - \bar{V}^*\right)(s_{t_i+1}) + \beta_t.$$

The proof is in Appendix A.6.

A.3 Proof of Theorem 4.1

Proof. Recall by our notation in Appendix A.1, $[\mathbb{P}_h V_{h+1}](s,a) = \mathbb{E}_{s'\sim\mathbb{P}_h(\cdot|s,a)} V_{h+1}(s')$ and its empirical counterpart of episode k is $\left[\hat{\mathbb{P}}_h V_{h+1}\right](s,a) = V_{h+1}(s_{h+1})$. By Lemma A.2, with probability at least $1-\delta$, we have $Q_h^k \ge Q_h^*$ and $Q_h^k \ge \bar{Q}_h^*$, i.e., $V_h^k \ge V_h^*$ and $V_h^k \ge \bar{V}_h^*$, thus

$$\text{Regret}(K) = \sum_{k=1}^{\bar{K}} (V_1^* - V_1^{\pi_k})(s_1^k) + \sum_{k=\bar{K}+1}^{K} (\bar{V}_1^* - \bar{V}_1^{\pi_k})(s_1^k) \tag{17}$$

$$\le \sum_{k=1}^{\bar{K}} (V_1^k - V_1^{\pi_k})(s_1^k) + \sum_{k=\bar{K}+1}^{K} (V_1^k - \bar{V}_1^{\pi_k})(s_1^k). \tag{18}$$

For any fixed $(k,h) \in \{1,\cdots,\bar{K}\}\times[H]$, let $\delta_h^k := \left(V_h^k - V_h^{\pi_k}\right)(s_1^k)$ and $\phi_h^k := \left(V_h^k - V_h^*\right)(s_h^k)$, then we have

$$\delta_h^k = \left(V_h^k - V_h^{\pi_k}\right)(s_1^k) \tag{19}$$

$$\le \left(Q_h^k - Q_h^{\pi_k}\right)(s_h^k, a_h^k) \left(\text{by } V_h^k(s_h^k) \le \max_{a'\in\mathcal{A}} Q_h^k(s_h^k, a') = Q_h^k(s_h^k, a_h^k)\right) \tag{20}$$

$$= \left(Q_h^k - Q_h^*\right)(s_h^k, a_h^k) + \left(Q_h^* - Q_h^{\pi_k}\right)(s_h^k, a_h^k) \tag{21}$$

$$\le \alpha_t^0 H + \sum_{i=1}^t \alpha_t^i \phi_{h+1}^{k_i} + \beta_t + \left[\mathbb{P}_h\left(V_{h+1}^* - V_{h+1}^{\pi_k}\right)\right](s_h^k, a_h^k) \text{ (by Lemma A.2)} \tag{22}$$

$$= \alpha_t^0 H + \sum_{i=1}^t \alpha_t^i \phi_{h+1}^{k_i} + \beta_t - \phi_{h+1}^k + \delta_{h+1}^k + \xi_{h+1}^k, \tag{23}$$

where $\beta_t = 2\sum_{i=1}^{t} \alpha_t^i b_i \leq 4\frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{H^3\iota/t} \leq 4\left(\frac{c}{p(\cdot|s,a;\theta_h)} + c\epsilon\right)\sqrt{H^3\iota/t}$ with $t = N_h(s,a)$, and $\xi_{h+1}^k := \left[(\mathbb{P}_h - \hat{\mathbb{P}}_h^k)(V_{h+1}^* - V_{h+1}^k)\right](s_h^k, a_h^k)$ is a martingale difference sequence.

Similarly, by Lemma A.2, for any fixed $(k,h) \in \{\bar{K}+1, \cdots, K\} \times [H]$, let $\bar{\delta}_h^k := \left(V_h^k - \bar{V}_h^{\pi_k}\right)(s_1^k)$ and $\bar{\phi}_h^k := \left(V_h^k - \bar{V}_h^*\right)(s_h^k)$, we have

$$\bar{\delta}_h^k = \alpha_t^0 H + \sum_{i=1}^{t} \alpha_t^i \bar{\phi}_{h+1}^{k_i} + \beta_t - \bar{\phi}_{h+1}^k + \bar{\delta}_{h+1}^k + \bar{\xi}_{h+1}^k, \tag{24}$$

where $\bar{\xi}_{h+1}^k := \left[(\bar{\mathbb{P}}_h - \hat{\mathbb{P}}_h^k)(\bar{V}_{h+1}^* - V_{h+1}^k)\right](s_h^k, a_h^k)$.

Hence, from Equation 17, the regret bound becomes

$$\text{Regret}(K) \leq \sum_{k=1}^{\bar{K}} \delta_h^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_h^k. \tag{25}$$

Denoting by $n_h^k = N_h^k(s_h^k, a_h^k)$, then we have

$$\sum_{k=1}^{\bar{K}} \alpha_{n_h^k}^0 H + \sum_{k=\bar{K}+1}^{K} \alpha_{n_h^k}^0 H = \sum_{k=1}^{K} \alpha_{n_h^k}^0 H = \sum_{k=1}^{K} H \cdot \mathbb{I}[n_h^k = 0] \leq |\mathcal{S}||\mathcal{A}|H. \tag{26}$$

Let $k_i(s_h^k, a_h^k)$ is the episode of which (s_h^k, a_h^k) was taken at step h for the $i-th$ time. For every $k' \in \{1, \cdots, \bar{K}\}$ and $k' \in \{\bar{K}+1, \cdots, K\}$, the corresponding term $\phi_{h+1}^{k'}$ and $\bar{\phi}_{h+1}^{k'}$ appears in the summand with $k > k'$ iff $(s_h^k, a_h^k) = (s_h^{k'}, a_h^{k'})$. The first time it appears we have $n_h^k = n_h^{k'} + 1$, the second time we have $n_h^k = n_h^{k'} + 2$, etc. Hence, we obtain the following bound

$$\sum_{k=1}^{\bar{K}} \sum_{i=1}^{n_h^k} \alpha_{n_h^k}^i \phi_{h+1}^{k_i(s_h^k, a_h^k)} + \sum_{k=\bar{K}+1}^{K} \sum_{i=1}^{n_h^k} \alpha_{n_h^k}^i \bar{\phi}_{h+1}^{k_i(s_h^k, a_h^k)} \leq \sum_{k'=1}^{\bar{K}} \phi_{h+1}^{k'} \sum_{t=n_h^{k'}+1}^{\infty} \alpha_t^{n_h^{k'}} + \sum_{k'=\bar{K}+1}^{K} \bar{\phi}_{h+1}^{k'} \sum_{t=n_h^{k'}+1}^{\infty} \alpha_t^{n_h^{k'}} \tag{27}$$

$$\leq \left(1 + \frac{1}{H}\right)\left(\sum_{k=1}^{\bar{K}} \phi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\phi}_{h+1}^k\right)\left(\text{by } \sum_{t=i}^{\infty} \alpha_t^i = 1 + \frac{1}{H} \text{ from Lemma A.1}\right). \tag{28}$$

Therefore, the regret bound becomes

$$\text{Regret}(K) \leq \sum_{k=1}^{\bar{K}} \delta_h^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_h^k \tag{29}$$

$$= \left(\sum_{k=1}^{\bar{K}} \alpha_t^0 H + \sum_{i=1}^{t} \alpha_t^i \phi_{h+1}^{k_i} + \beta_t - \phi_{h+1}^k + \delta_{h+1}^k + \xi_{h+1}^k\right) + \left(\sum_{k=\bar{K}+1}^{K} \alpha_t^0 H + \sum_{i=1}^{t} \alpha_t^i \bar{\phi}_{h+1}^{k_i} + \beta_t - \bar{\phi}_{h+1}^k + \bar{\delta}_{h+1}^k + \bar{\xi}_{h+1}^k\right) \tag{30}$$

$$\leq |\mathcal{S}||\mathcal{A}|H + \left(1 + \frac{1}{H}\right)\left(\sum_{k=1}^{\bar{K}} \phi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\phi}_{h+1}^k\right) - \left(\sum_{k=1}^{\bar{K}} \phi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\phi}_{h+1}^k\right) + \left(\sum_{k=1}^{\bar{K}} \delta_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_{h+1}^k\right) + \left(\sum_{k=1}^{\bar{K}} \beta_{n_h^k} + \sum_{k=\bar{K}+1}^{K} \beta_{n_h^k}\right) + \left(\sum_{k=1}^{\bar{K}} \xi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\xi}_{h+1}^k\right) \tag{31}$$

$$\leq |\mathcal{S}||\mathcal{A}|H + \left(1 + \frac{1}{H}\right)\left(\sum_{k=1}^{\bar{K}} \phi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\phi}_{h+1}^k\right) + \left(\sum_{k=1}^{\bar{K}} \delta_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_{h+1}^k\right) + \sum_{k=1}^{K} \beta_{n_h^k} + \left(\sum_{k=1}^{\bar{K}} \xi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\xi}_{h+1}^k\right). \tag{32}$$

Combining with the fact that $V^* \geq V^{\pi_k}$ and $\bar{V}^* \geq \bar{V}^{\pi_k}$, yielding $\phi_{h+1}^k \leq \delta_{h+1}^k$ and $\bar{\phi}_{h+1}^k \leq \bar{\delta}_{h+1}^k$, thus we get

$$\text{Regret}(K) \leq |\mathcal{S}||\mathcal{A}|H + \left(1 + \frac{1}{H}\right)\left(\sum_{k=1}^{\bar{K}} \delta_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_{h+1}^k\right) + \sum_{k=1}^{K} \beta_{n_h^k} + \left(\sum_{k=1}^{\bar{K}} \xi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\xi}_{h+1}^k\right). \tag{33}$$

Recursing the result for $h = 1, 2, \cdots, H$, and using the fact that $\delta_{H+1}^K = 0$, we have

$$\text{Regret}(K) \leq \mathcal{O}\left(H^2|\mathcal{S}||\mathcal{A}| + \sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{n_h^k} + \sum_{h=1}^{H}\left(\sum_{k=1}^{\bar{K}} \xi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\xi}_{h+1}^k\right)\right). \tag{34}$$

On the one hand, the r.v. $\xi_{h+1}^k := \left[(\mathbb{P}_h - \hat{\mathbb{P}}_h^k)(V_{h+1}^* - V_{h+1}^k)\right](s_h^k, a_h^k)$ is a martingale difference sequence because if we define $\mathcal{F}_i$ be the $\sigma-$field generated by all the r.v. until episode k_i, step h, then, ξ_{h+1}^k is a martingale difference sequence w.r.t the filtration $\{\mathcal{F}_i\}_{i\geq 0}$. Hence, by the Azuma-Hoeffding inequality, with probability $1-\delta$, we have

$$\begin{aligned}\left|\sum_{h=1}^{H}\left(\sum_{k=1}^{\bar{K}} \xi_{h+1}^k + \sum_{k=\bar{K}+1}^{K} \bar{\xi}_{h+1}^k\right)\right| &= \left|\sum_{h=1}^{H}\left(\sum_{k=1}^{\bar{K}} \left[(\mathbb{P}_h - \hat{\mathbb{P}}_h^k)(V_{h+1}^* - V_{h+1}^k)\right](s_h^k, a_h^k)\right.\right. \\ &\quad \left.\left. + \sum_{k=\bar{K}+1}^{K} \left[(\bar{\mathbb{P}}_h - \hat{\mathbb{P}}_h^k)(\bar{V}_{h+1}^* - V_{h+1}^k)\right](s_h^k, a_h^k)\right)\right| \qquad (35) \\ &\leq \frac{c}{\min_{h\in[H]} p(\cdot|s,a;\theta_h)} H\sqrt{KH\iota} \leq \left(\frac{c}{\min_{h\in[H]} p(\cdot|s,a;\theta_h)} + c\epsilon\right) H\sqrt{KH\iota}. \qquad (36)\end{aligned}$$

On the other hand, by the pigeonhole principle,

$$\sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{n_h^k} \leq \sum_{h=1}^{H} \mathcal{O}(1+\epsilon) \sum_{k=1}^{K} \sqrt{\frac{H^3\iota}{n_h^k}} = \sum_{h=1}^{H} \mathcal{O}(1+\epsilon) \sum_{s,a} \sum_{n=1}^{N_h^K(s,a)} \sqrt{\frac{H^3\iota}{n}}, \tag{37}$$

combining with the fact that $\sum_{s,a} N_h^K(s,a) = K$ and $\sum_{s,a}\sum_{n=1}^{N_h^K(s,a)} \sqrt{\frac{H^3\iota}{n}}$ is maximized when $N_h^K(s,a) = K/|\mathcal{S}||\mathcal{A}|$ for all s, a, we get

$$\sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{n_h^k} \leq \mathcal{O}\left(\sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|K\iota}\right). \tag{38}$$

Note that when $KH \geq \sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|T\iota}$, by infinitely nested radical, we have $\sqrt{H^5(1+\epsilon)^2\mathcal{S}||\mathcal{A}|K\iota} \geq H^2|\mathcal{S}||\mathcal{A}|$. And, when $KH \leq \sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|K\iota}$, by the worst case of regret is KH, we have $\sum_{k=1}^{\bar{K}} \delta_h^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_h^k \leq KH \leq \sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|K\iota}$. Therefore, by $\iota = \log(|\mathcal{S}||\mathcal{A}|KH/\delta)$, with probability at least $1-\delta$, we obtain

$$\text{Regret}(K) \leq \sum_{k=1}^{\bar{K}} \delta_h^k + \sum_{k=\bar{K}+1}^{K} \bar{\delta}_h^k \leq \mathcal{O}\left(\sqrt{H^5(1+\epsilon)^2|\mathcal{S}||\mathcal{A}|K\log(|\mathcal{S}||\mathcal{A}|KH/\delta)}\right) \tag{39}$$

of Theorem 4.1. □

A.4 Proof of Theorem 4.3

Proof. Similar to the proof in Theorem 4.1, this proof is based on the bound on learning error of Q-function in the infinite-horizon discounted MDP setting. Firstly, from Lemma A.3, we can see that the different between the bound of $(\hat{Q}^t - Q^*)(s,a)$ and $(\hat{Q}^t - \bar{Q}^*)(s,a)$ is only in V^* and $\bar{V}^*$. Therefore, we can borrow Theorem 7.3

of Yang et al. (2021), i.e., with probability at least $1-\delta$, where $\delta = 1/T$, for every $n \in [\lceil \log_2(1/\Delta_{\min}(1-\gamma))\rceil]$, we have

$$C^{(n)} := \left|\left\{t \in \mathbb{N}_+ : \left(\hat{Q}^t - Q^*\right)(s_t, a_t) \in \left[2^{n-1}\Delta_{\min}, 2^n \Delta_{\min}\right]\right\}\right| \tag{40}$$

$$\leq \mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{4^n(1-\gamma)^5\Delta_{\min}^2} \ln\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\Delta_{\min}}\right)\right), \tag{41}$$

and for every $n \in [\lceil \log_2(1/\bar{\Delta}_{\min}(1-\gamma))\rceil]$,

$$\bar{C}^{(n)} := \left|\left\{t \in \mathbb{N}_+ : \left(\hat{Q}^t - \bar{Q}^*\right)(s_t, a_t) \in \left[2^{n-1}\bar{\Delta}_{\min}, 2^n \bar{\Delta}_{\min}\right]\right\}\right| \tag{42}$$

$$\leq \mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{4^n(1-\gamma)^5\bar{\Delta}_{\min}^2} \ln\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\bar{\Delta}_{\min}}\right)\right). \tag{43}$$

By the definition of the sub-optimality gap, we have

$$\text{Regret}(T) = \sum_{t=1}^{\bar{T}} \left(V^* - V^{\pi_t}\right)(s_t) + \sum_{t=\bar{T}+1}^{T} \left(\bar{V}^* - \bar{V}^{\pi_t}\right)(s_t) \tag{44}$$

$$= \sum_{t=1}^{\bar{T}} \mathbb{E}\left[\sum_{h=0}^{\infty} \gamma^h \Delta(s_{t+h}, a_{t+h}) \middle| a_{t+h} = \pi_{t+h}(s_{t+h})\right]$$

$$+ \sum_{t=\bar{T}+1}^{T} \mathbb{E}\left[\sum_{h=0}^{\infty} \gamma^h \bar{\Delta}(s_{t+h}, a_{t+h}) \middle| a_{t+h} = \pi_{t+h}(s_{t+h})\right] \tag{45}$$

$$= \sum_{t=1}^{\bar{T}} \mathbb{E}\left[\sum_{h=0}^{\infty} \gamma^h \Delta(s_{t+h}, a_{t+h})\right] + \sum_{t=\bar{T}+1}^{T} \mathbb{E}\left[\sum_{h=0}^{\infty} \gamma^h \bar{\Delta}(s_{t+h}, a_{t+h})\right] \tag{46}$$

$$= \sum_{t=1}^{\bar{T}} \sum_{h'=t}^{\infty} \gamma^{h'-t} \Delta(s_{h'}, a_{h'}) + \sum_{t=\bar{T}+1}^{T} \sum_{h'=t}^{\infty} \gamma^{h'-t} \bar{\Delta}(s_{h'}, a_{h'}). \tag{47}$$

For a fixed s_t, for every infinite-length trajectory $traj$, the trajectories inside the event in which all the learning errors of the value function is both bounded below (by zero) and bounded above, we can apply the concentration bound with Azuma–Hoeffding to the sub-optimality gap (Yang et al., 2021), whereas for trajectories outside of this event, the sub-optimality gaps never exceed the time horizon. Hence, by the Azuma–Hoeffding inequality, with probability at least $1-\delta$, we have

$$\text{Regret}(T) \leq \sum_{t=1}^{\bar{T}} \sum_{traj} p(traj) \left[\sum_{h'=t}^{\infty} \gamma^{h'-t} \Delta(s_{h'}, a_{h'})\right] + \sum_{t=\bar{T}+1}^{T} \sum_{traj} p(traj) \left[\sum_{h'=t}^{\infty} \gamma^{h'-t} \bar{\Delta}(s_{h'}, a_{h'})\right] \tag{48}$$

$$= \sum_{t=1}^{\bar{T}} \sum_{h'=t}^{\infty} \gamma^{h'-t} \Delta(s_{h'}, a_{h'}|traj) + \sum_{t=\bar{T}+1}^{T} \sum_{h'=t}^{\infty} \gamma^{h'-t} \bar{\Delta}(s_{h'}, a_{h'}|traj) \tag{49}$$

$$= \sum_{h=1}^{\infty} \Delta(s_h, a_h) \sum_{t=1}^{\min\{\bar{T},h\}} \gamma^t + \sum_{h=1}^{\infty} \bar{\Delta}(s_h, a_h) \sum_{t=\bar{T}+1}^{\min\{T,h\}} \gamma^t \tag{50}$$

$$\leq 2 \max\left\{\frac{1}{1-\gamma} \sum_{h=1}^{\infty} \Delta(s_h, a_h|traj), \frac{1}{1-\gamma} \sum_{h=1}^{\infty} \bar{\Delta}(s_h, a_h|traj)\right\} \tag{51}$$

(By the optimism of estimated Q-values).

Since we can add an outer summation over sub-intervals $n \in [N]$ and bound each of them by their maximum value times the number of steps inside, we get

$$\text{Regret}(T) \leq 2 \max\left\{\frac{1}{1-\gamma} \sum_{n=1}^{N} 2^n \Delta_{\min} C^{(n)}, \frac{1}{1-\gamma} \sum_{n=1}^{N} 2^n \bar{\Delta}_{\min} \bar{C}^{(n)}\right\}. \tag{52}$$

Using results in Equation 40 and Equation 42, we obtain

$$\text{Regret}(T) \leq \max\left\{\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{(1-\gamma)^6\Delta_{\min}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\Delta_{\min}}\right)\right), \mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{(1-\gamma)^6\bar{\Delta}_{\min}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\bar{\Delta}_{\min}}\right)\right)\right\} \tag{53}$$

$$= \mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|(1+\epsilon)}{(1-\gamma)^6\min\{\Delta_{\min},\bar{\Delta}_{\min}\}}\log\left(\frac{|\mathcal{S}||\mathcal{A}|T}{(1-\gamma)\min\{\Delta_{\min},\bar{\Delta}_{\min}\}}\right)\right). \tag{54}$$

of Theorem 4.3. □

A.5 Proof of Lemma A.2

Proof. For any $(s,a,h) \in \mathcal{S}\times\mathcal{A}\times[H]$ and episode $k \in [K]$, from the Bellman optimality equation, $Q_h^*(s,a) = (r_h + \mathbb{P}_h V_{h+1}^*)(s,a)$ and $\bar{Q}_h^*(s,a) = (r_h + \bar{\mathbb{P}}_h \bar{V}_{h+1}^*)(s,a)$, since $\left[\hat{\mathbb{P}}_h^{k_i} V_{h+1}\right](s,a) = V_{h+1}(s_{h+1}^{k_i})$, we have $\left[\hat{\mathbb{P}}_h^{k_i} \bar{V}_{h+1}\right](s,a) = \bar{V}_{h+1}(s_{h+1}^{k_i})$, thus

$$Q_h^*(s,a) = \alpha_t^0 Q_h^*(s,a) + \sum_{i=1}^t \alpha_t^i\left[r_h(s,a) + (\mathbb{P}_h - \hat{\mathbb{P}}_h^{k_i})V_{h+1}^*(s,a) + V_{h+1}^*(s_{h+1}^{k_i})\right], \tag{55}$$

$$\bar{Q}_h^*(s,a) = \alpha_t^0 \bar{Q}_h^*(s,a) + \sum_{i=1}^t \alpha_t^i\left[r_h(s,a) + (\bar{\mathbb{P}}_h - \hat{\mathbb{P}}_h^{k_i})\bar{V}_{h+1}^*(s,a) + \bar{V}_{h+1}^*(s_{h+1}^{k_i})\right]. \tag{56}$$

Subtracting to $Q_h^k(s,a)$ in Equation 13, we obtain

$$(Q_h^k - Q_h^*)(s,a) = \alpha_t^0(H - Q_h^*(s,a)) + \sum_{i=1}^t \alpha_t^i\left[(V_{h+1}^{k_i} - V_{h+1}^*)(s_{h+1}^{k_i}) + \left[(\hat{\mathbb{P}}_h^{k_i} - \mathbb{P}_h)V_{h+1}^*\right](s,a) + b_i\right], \tag{57}$$

$$(Q_h^k - \bar{Q}_h^*)(s,a) = \alpha_t^0(H - \bar{Q}_h^*(s,a)) + \sum_{i=1}^t \alpha_t^i\left[(V_{h+1}^{k_i} - \bar{V}_{h+1}^*)(s_{h+1}^{k_i}) + \left[(\hat{\mathbb{P}}_h^{k_i} - \bar{\mathbb{P}}_h)\bar{V}_{h+1}^*\right](s,a) + b_i\right]. \tag{58}$$

If $t = N_h^k(s,a) \in \{1,\cdots,\bar{K}-1\}$, the transition operator at step h is $\mathbb{P}_h$, then $p(\cdot|s,a;\theta_h) = \mathbb{P}_h/\mathbb{P}_h$, and for all $(s,a,h,k) \in \mathcal{S}\times\mathcal{A}\times[H]\times\{1,\cdots,\bar{K}-1\}$, from Lemma 4.3 (Jin et al., 2018), with probability at least $1-\delta$, we have

$$\left|\sum_{i=1}^t \alpha_t^i\left[(\hat{\mathbb{P}}_h^{k_i} - \mathbb{P}_h)V_{h+1}^*\right](s,a)\right| \leq c\frac{\mathbb{P}_h}{\mathbb{P}_h}\sqrt{\frac{H^3\iota}{t}} = \frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{\frac{H^3\iota}{t}}. \tag{59}$$

Consider the transition operator at step h change from $\mathbb{P}_h$ to $\bar{\mathbb{P}}_h$ at episode $\bar{K}$, then $p(\cdot|s,a;\theta_h) = \mathbb{P}_h/\mathbb{P}_h$ before $\bar{K}$, $p(\cdot|s,a;\theta_h) = \mathbb{P}_h/\bar{\mathbb{P}}_h$ at $\bar{K}$, and for $t = \bar{K}$, $\forall(s,a,h,k) \in \mathcal{S}\times\mathcal{A}\times[H]\times\{\bar{K}\}$, we have

$$\left|\sum_{i=1}^t \alpha_t^i\left[(\hat{\mathbb{P}}_h^{k_i} - \bar{\mathbb{P}}_h)\bar{V}_{h+1}^*\right](s,a)\right| = \left|\sum_{i=1}^t \alpha_t^i\left[(\hat{\mathbb{P}}_h^{k_i} - \mathbb{P}_h)V_{h+1}^*\right](s,a)\right|\frac{\bar{\mathbb{P}}_h}{\mathbb{P}_h}. \tag{60}$$

Hence, using the result in Equation 59, we get

$$\left|\sum_{i=1}^t \alpha_t^i\left[(\hat{\mathbb{P}}_h^{k_i} - \bar{\mathbb{P}}_h)\bar{V}_{h+1}^*\right](s,a)\right| \leq c\frac{\mathbb{P}_h}{\mathbb{P}_h}\frac{\bar{\mathbb{P}}_h}{\mathbb{P}_h}\sqrt{\frac{H^3\iota}{t}} = c\frac{\bar{\mathbb{P}}_h}{\mathbb{P}_h}\sqrt{\frac{H^3\iota}{t}} = \frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{\frac{H^3\iota}{t}}. \tag{61}$$

And, when the transition operator at step h is $\bar{\mathbb{P}}_h$ after $\bar{K}$, then $p(\cdot|s,a;\theta_h) = \bar{\mathbb{P}}_h/\bar{\mathbb{P}}_h$, and for any $t \in \{\bar{K}+1,\cdots,K\}$, $\forall(s,a,h,k) \in \mathcal{S}\times\mathcal{A}\times[H]\times\{\bar{K}+1,\cdots,K\}$, with probability at least $1-\delta$, we have

$$\left|\sum_{i=\bar{K}+1}^t \alpha_t^i\left[(\hat{\mathbb{P}}_h^{k_i} - \bar{\mathbb{P}}_h)\bar{V}_{h+1}^*\right](s,a)\right| \leq c\frac{\bar{\mathbb{P}}_h}{\bar{\mathbb{P}}_h}\sqrt{\frac{H^3\iota}{t}} = \frac{c}{p(\cdot|s,a;\theta_h)}\sqrt{\frac{H^3\iota}{t}}. \tag{62}$$

Combining the result in Equation 59, Equation 61, and Equation 62, we obtain for any $t \in [K]$, $\forall (s,a,h,k) \in \mathcal{S} \times \mathcal{A} \times [H] \times [K]$, with probability at least $1-\delta$, it holds that

$$\left| \sum_{i=1}^{t} \alpha_t^i \left[(\hat{\mathbb{P}}_h^{k_i} - \mathbb{P}_h) V_{h+1}^* \right](s,a) \right| \leq \frac{c}{p(\cdot|s,a;\theta_h)} \sqrt{\frac{H^3 \iota}{t}}, \text{ where } t = N_h^k(s,a). \tag{63}$$

Note that if we choose $b_t = \frac{c}{p(\cdot|s,a;\theta_h)} \sqrt{H^3\iota/t}$, i.e., $\beta_t = 2\sum_{i=1}^t \alpha_t^i b_i \leq 4 \frac{c}{p(\cdot|s,a;\theta_h)} \sqrt{H^3\iota/t}$, by Lemma A.1, $\beta_t/2 = \sum_{i=1}^t \alpha_t^i b_i \in \left[\frac{c}{p(\cdot|s,a;\theta_h)} \sqrt{\frac{H^3\iota}{t}}, \frac{2c}{p(\cdot|s,a;\theta_h)} \sqrt{\frac{H^3\iota}{t}} \right]$, combining with results in Equation 57 and Equation 63, we obtain

$$(Q_h^k - Q_h^*)(s,a) \leq \alpha_t^0 H + \sum_{i=1}^{t} \alpha_t^i \left(V_{h+1}^{k_i} - V_{h+1}^* \right)(s_{h+1}^{k_i}) + \beta_t, \tag{64}$$

$$(Q_h^k - \bar{Q}_h^*)(s,a) \leq \alpha_t^0 H + \sum_{i=1}^{t} \alpha_t^i \left(V_{h+1}^{k_i} - \bar{V}_{h+1}^* \right)(s_{h+1}^{k_i}) + \beta_t. \tag{65}$$

On the other hand, applying induction on $h = H, H-1, \cdots, 1$ to results in Equation 57 and result in Equation 63, we obtain

$$0 \leq (Q_h^k - Q_h^*)(s,a), \quad 0 \leq (Q_h^k - \bar{Q}_h^*)(s,a). \tag{66}$$

Combining results in Equation 64 and Equation 66, we obtain the results of Lemma A.2. □

A.6 Proof of Lemma A.3

Proof. For any $(s,a) \in \mathcal{S} \times \mathcal{A}$ and time step $t \in [T]$, from the Bellman optimality equation, we have

$$Q^*(s,a) = r(s,a) + \gamma \mathbb{P} V^*(s,a) = \alpha_t^0 Q^*(s,a) + \sum_{t=1}^{t} \alpha_t^i \left[r(s,a) + \gamma \mathbb{P} V^*(s,a) \right], \tag{67}$$

$$\bar{Q}^*(s,a) = r(s,a) + \gamma \bar{\mathbb{P}} \bar{V}^*(s,a) = \alpha_t^0 \bar{Q}^*(s,a) + \sum_{t=1}^{t} \alpha_t^i \left[r(s,a) + \gamma \bar{\mathbb{P}} \bar{V}^*(s,a) \right]. \tag{68}$$

Subtracting to $\hat{Q}^t(s,a)$ in Equation 15, we obtain

$$\begin{aligned} \left(\hat{Q}^t - Q^* \right)(s,a) &= \alpha_t^0 \left(\frac{1}{1-\gamma} - Q^*(s,a) \right) \\ &\quad + \sum_{i=1}^{t} \alpha_t^i \left[\gamma (V_{t_1} - V^*)(s_{t_i+1}) + \gamma \left(V^*(s_{t_i+1}) - \mathbb{P} V^*(s,a) \right) + b_i \right], \end{aligned} \tag{69}$$

$$\begin{aligned} \left(\hat{Q}^t - \bar{Q}^* \right)(s,a) &= \alpha_t^0 \left(\frac{1}{1-\gamma} - \bar{Q}^*(s,a) \right) \\ &\quad + \sum_{i=1}^{t} \alpha_t^i \left[\gamma (V_{t_1} - \bar{V}^*)(s_{t_i+1}) + \gamma \left(\bar{V}^*(s_{t_i+1}) - \bar{\mathbb{P}} \bar{V}^*(s,a) \right) + b_i \right], \end{aligned} \tag{70}$$

If $t = N_p(s,a) \in \{1, \cdots, \bar{T}-1\}$, $t_i = \tau(s,a,i)$, the transition operator at step t is $\mathbb{P}$, then $p(\cdot|s,a;\theta) = \mathbb{P}/\mathbb{P}$, and for all $(s,a,t) \in \mathcal{S} \times \mathcal{A} \times \{1, \cdots, \bar{T}-1\}$, from Lemma 4 (Wang et al., 2020), with probability at least $1-\delta$, we have

$$\gamma \left| \sum_{i=1}^{t} \left(\alpha_k^i \mathbb{I}[t_i < \infty] \left(\mathbb{P} - \hat{\mathbb{P}}_{t_i} \right) V^*(s,a) \right) \right| \leq \frac{3c_2}{1-\gamma} \frac{\mathbb{P}}{\mathbb{P}} \sqrt{\frac{H\iota(t)}{t}} = \frac{3c_2}{(1-\gamma)p(\cdot|s,a;\theta)} \sqrt{\frac{H\iota(t)}{t}}. \tag{71}$$

Consider the transition operator at step t change from $\mathbb{P}$ to $\bar{\mathbb{P}}$ at time step $\bar{T}$, then $p(\cdot|s,a;\theta) = \mathbb{P}/\mathbb{P}$ before $\bar{T}$, $p(\cdot|s,a;\theta) = \mathbb{P}/\bar{\mathbb{P}}$ at $\bar{T}$, and for any $t = \bar{T}$, $\forall (s,a,t) \in \mathcal{S} \times \mathcal{A} \times \{\bar{T}\}$, we have

$$\gamma \left| \sum_{i=1}^{t} \left(\alpha_k^i \mathbb{I}[t_i < \infty] \left(\bar{\mathbb{P}} - \hat{\mathbb{P}}_{t_i} \right) \bar{V}^*(s,a) \right) \right| = \gamma \left| \sum_{i=1}^{t} \left(\alpha_k^i \mathbb{I}[t_i < \infty] \left(\mathbb{P} - \hat{\mathbb{P}}_{t_i} \right) V^*(s,a) \right) \right| \frac{\bar{\mathbb{P}}}{\mathbb{P}}. \tag{72}$$

Hence, using the result in Equation 71, we get

$$\gamma\left|\sum_{i=1}^{t}\left(\alpha_k^i\mathbb{I}[t_i<\infty]\left(\bar{\mathbb{P}}-\hat{\mathbb{P}}_{t_i}\right)\bar{V}^*(s,a)\right)\right| \leq \frac{3c_2}{(1-\gamma)}\frac{\mathbb{P}}{\mathbb{P}}\frac{\bar{\mathbb{P}}}{\mathbb{P}}\sqrt{\frac{H\iota(t)}{t}} = \frac{3c_2}{(1-\gamma)}\frac{\bar{\mathbb{P}}}{\mathbb{P}}\sqrt{\frac{H\iota(t)}{t}} \tag{73}$$

$$= \frac{3c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}}. \tag{74}$$

And, when the transition operator at step t is $\bar{\mathbb{P}}$ after $\bar{T}$, then $p(\cdot|s,a;\theta)=\bar{\mathbb{P}}/\bar{\mathbb{P}}$, and for any $t\in\{\bar{T}+1,\cdots,T\}$, $\forall(s,a,t)\in\mathcal{S}\times\mathcal{A}\times\{\bar{T}+1,\cdots,T\}$, with probability at least $1-\delta$, we have

$$\gamma\left|\sum_{i=\bar{T}+1}^{t}\left(\alpha_k^i\mathbb{I}[t_i<\infty]\left(\bar{\mathbb{P}}-\hat{\mathbb{P}}_{t_i}\right)\bar{V}^*(s,a)\right)\right| \leq \frac{3c_2}{1-\gamma}\frac{\bar{\mathbb{P}}}{\bar{\mathbb{P}}}\sqrt{\frac{H\iota(t)}{t}} = \frac{3c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}}. \tag{75}$$

Combining the result in Equation 71, Equation 73, and Equation 75, we obtain for any $t\in[K]$, $\forall(s,a,t)\in\mathcal{S}\times\mathcal{A}\times[T]$, with probability at least $1-\delta$, it holds that

$$\gamma\left|\sum_{i=1}^{t}\left(\alpha_k^i\mathbb{I}[t_i<\infty]\left(\mathbb{P}-\hat{\mathbb{P}}_{t_i}\right)V^*(s,a)\right)\right| \leq \frac{3c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}},\ \text{where } t=N_p(s,a). \tag{76}$$

Note that if we choose $b_t=\frac{c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}}$, i.e., $\beta_t=2\sum_{i=1}^t\alpha_t^ib_i\leq 4\frac{c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}}$, by Lemma A.1, $\beta_t/2=\sum_{i=1}^t\alpha_t^ib_i\in\left[\frac{c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}},\frac{2c_2}{(1-\gamma)p(\cdot|s,a;\theta)}\sqrt{\frac{H\iota(t)}{t}}\right]$, combining with results in Equation 69 and Equation 76, we obtain

$$\begin{aligned}(\hat{Q}^t-Q^*)(s,a) &\leq \frac{\alpha_t^0}{1-\gamma}+\gamma\left|\sum_{i=1}^{t}\left(\alpha_k^i\mathbb{I}[t_i<\infty]\left(\mathbb{P}-\hat{\mathbb{P}}_{t_i}\right)V^*(s,a)\right)\right| \\ &\quad+\sum_{i=1}^{t}\alpha_t^i\left[\gamma(\hat{V}_{t_1}-V^*)(s_{t_i+1})+b_i\right] \qquad (77)\\ &\leq \frac{\alpha_t^0}{1-\gamma}+\sum_{i=1}^{t}\gamma\alpha_t^i\left(\hat{V}_{t_i}-V^*\right)(s_{t_i+1})+\beta_t, \qquad (78)\end{aligned}$$

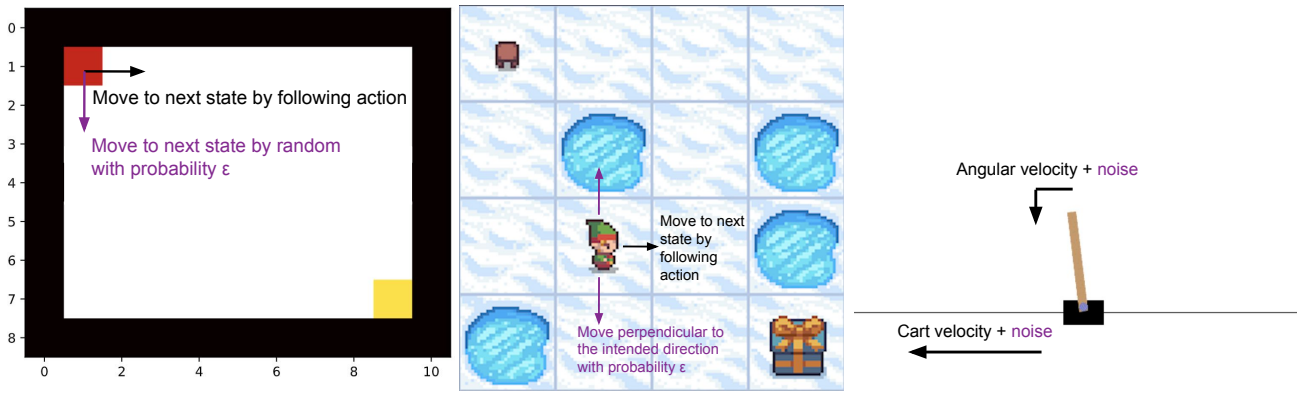
and

$$\begin{aligned}(\hat{Q}^t-\bar{Q}^*)(s,a) &\leq \frac{\alpha_t^0}{1-\gamma}+\gamma\left|\sum_{i=1}^{t}\left(\alpha_k^i\mathbb{I}[t_i<\infty]\left(\bar{\mathbb{P}}-\hat{\mathbb{P}}_{t_i}\right)\bar{V}^*(s,a)\right)\right| \\ &\quad+\sum_{i=1}^{t}\alpha_t^i\left[\gamma(\hat{V}_{t_1}-\bar{V}^*)(s_{t_i+1})+b_i\right] \qquad (79)\\ &\leq \frac{\alpha_t^0}{1-\gamma}+\sum_{i=1}^{t}\gamma\alpha_t^i\left(\hat{V}_{t_i}-\bar{V}^*\right)(s_{t_i+1})+\beta_t. \qquad (80)\end{aligned}$$

On the other hand, applying induction on $t=T,T-1,\cdots,1$ to results in Equation 69 and result in Equation 76, we obtain

$$0\leq(\hat{Q}^t-Q^*)(s,a),\quad 0\leq(\hat{Q}^t-\bar{Q}^*)(s,a). \tag{81}$$

Combining results in Equation 77 and Equation 81, we obtain the results of Lemma A.3. □



(a) (b) (c)

Figure 7: (a) The GridWorld task: the starting state is shown in red, the rewarding state is shown in yellow, and the transitions are noisy with ϵ; (b) The Frozen-Lake task: the starting state is the chair at the top-left corner, the rewarding state is the box at the bottom-right corner, the goal is crossing a frozen lake from start to the rewarding state without falling into any holes by walking over the frozen lake, and the transitions are noisy with a slippery level ϵ; (c) The CartPole task: the pendulum is placed upright on the cart, the goal is to balance the pole, and the transition noise $\mathcal{N}(0, 0.15)$ is added to the velocity state.

B Experimental details

B.1 Experimental settings

B.1.1 GridWorld

Fig. 7 (a) shows our $[10] \times [5]$ GridWorld environment with 50 states and 4 actions (left, right, up, down). When the agent uses its policy to take action, it will move in the corresponding direction following the action with probability $1 - \epsilon$, and move to a neighbor state at random with probability ϵ. The starting position is $(1, 1)$. The reward is equal to 1 in state $(10, 5)$ and is zero elsewhere. We set the planning horizon $H = 100$ and the number of episodes $K = 50000$. From the first episode to $\bar{K} = 25000$, the noise is $\epsilon = 0.01$. After that, the transition function is shifted by changing $\epsilon = 0.2$.

State: The state $s_t = (i, j) \in [10] \times [5]$ at step t consists of the agent's position on the grid, where i is the row and j is the column index. **Action**: Action $a_t \in \{left, right, up, down\}$ at time t is the direction in which the agent will try to reach. **State Transition Function**: From position state s_t, the next position of agent s_{t+1} will follow action a_t with probability $1 - \epsilon$, and move to a random neighbor position of s_t with probability ϵ. **Reward Function**: The reward r_t equals 1 when the agent reaches the final state $(10, 5)$, and equals 0 elsewhere.

Baselines: We compare our method with: (1) Q-learning UCB (QUCB) (Jin et al., 2018); (2) Q-learning UCB, but additionally incorporates a momentum term that is built from the value functions for each state-action pair (UCBMQ) (Menard et al., 2021); (3) a model-based RL baseline which directly adds a UCB bonus to the Q-values (UCBVI) Azar et al. (2017).

B.1.2 Frozen-Lake

Fig. 7 (c) shows our $[4] \times [4]$ FrozenLake environment with 16 states and 4 actions (left, right, up, down). This is a more challenging setting than GridWorld, as it involves crossing a frozen lake from the start to the goal without falling into any holes by walking across the frozen lake. When the agent uses its policy to take action, it will move in the corresponding direction following the action with probability $1 - \epsilon$, and move in either perpendicular direction with equal probability (i.e., $\epsilon/2$) in both directions. The starting position is $(1, 1)$. The reward is equal to 1 in state $(4, 4)$ and is zero elsewhere. The episode ends if the agent moves to the hole or reaches the goal. We set the planning horizon $H = 500$ and the number of episodes $K = 60000$. From the first episode to $\bar{K} = 20000$, the noise is $\epsilon = 0$. Then, from episode 20000-th to $\bar{K} = 40000$, the noise $\epsilon = 1/2$. After that, the transition function is shifted by changing $\epsilon = 2/3$.

State: The state $s_t = (i, j) \in [8] \times [8]$ at step t consists of the agent's position on the grid, where i is the row and j is the column index. **Action**: Action $a_t \in \{left, right, up, down\}$ at time t is the direction in which the agent will try to reach. **State Transition Function**: From position state s_t, the next position of the agent s_{t+1} will follow action a_t with probability $1 - \epsilon$, and move in either perpendicular direction with equal probability (i.e., $\epsilon/2$) in both directions. **Reward Function**: The reward r_t equals 1 when the agent reaches the final state $(8, 8)$, and equals 0 elsewhere.

Baselines: Similarly to the GridWorld task, we compare our method with the following tabular UCB-based RL baselines: (1) QUCB (Jin et al., 2018); (2) UCBMQ (Menard et al., 2021); (3) UCBVI Azar et al. (2017).

B.1.3 CartPole

Fig. 7 (c) shows this task with 4 continuous state features (cart position, cart velocity, pole angle, pole angular velocity), and 2 actions (left, right). The goal of each episode is to keep the pole upright as long as possible, and the reward is +1 for each step taken. We set the number of episodes $K = 800$. After episode $\bar{K} = 400$, the shift occurs by adding Gaussian noise $\mathcal{N}(0, 0.15)$ to the cart and pole angular velocity features.

State: The state $s_t \in \mathbb{R}^4$ at time t consists the following positions and velocities: (1) the cart position can take values between $(-4.8, 4.8)$, but the episode terminates if the cart leaves the $(-2.4, 2.4)$ range; (2) the cart velocity can take values between $(-\infty, \infty)$; (3) the pole angle can be observed between $(-0.418, 0.148)$ radians, but the episode terminates if the pole angle is not in the range $(-0.2095, 0.2095)$; (4) the pole angular velocity can take values between $(-\infty, \infty)$.

Action: The action $a_t \in \{0, 1\}$ at time t indicates the direction of the fixed force with which the cart is driven, where 0 represents pushing the cart to the left and 1 represents pushing the cart to the right.

State Transition Function: From the state s_t, the cart changed its positions and velocities to state s_{t+1} by action a_t, with additional Gaussian noise $\epsilon \sim \mathcal{N}(0, 0.15)$ to the cart and pole angular velocity features. All observations are assigned a uniformly random value in $(-0.05, 0.05)$. The episode ends if any one of the following occurs: (1) Termination: the pole angle is over ± 0.2095 or the cart position is over ± 2.4 (center of the cart reaches the edge of the display); (2) Truncation: episode length is greater than 200.

Reward Function: Since the goal is to keep the pole upright as long as possible, a reward r_t is added by +1 for step t, including the termination step. The reward threshold is 195.

Baselines: We compare our method with: (1) Random policy; (2) Deep Q-Network (DQN) (Mnih et al., 2013; Paszke et al., 2019); (3) Deep Q-Network with UCB exploration (DQN-UCB). We adapt the hashing technique of Tang et al. (2017) to count the number of visited pairs $N_h(s, a)$ on this continuous state space. In particular, count-based exploration uses a static hashing to map continuous states into discrete states and then counts the number of times a given state has been visited. After that, the UCB algorithms are trained with a bonus reward considering the number of times we have visited the state. This bonus reward plays the role of exploration.

B.1.4 COVID-19 patient hospital allocation

Notation:

- K: Number of hospitals.
- T: Number of days in the planning horizon.
- c_i: Capacity of hospital i, for $i = 1, \cdots, K$.
- N_t: Number of arriving COVID-19 patients in the system on day t.
- $\{A_t\}_{t \in [T]}$: Stochastic process representing the number of arriving COVID-19 patients over time.
- $d_{i,t}$: Number of non-COVID patients in hospital i on day t.
- $\{D_{i,t}\}_{t \in [T]}$: Stochastic process representing the number of non-COVID patients in hospital i over time.
- $a_{i,t}$ (action): Number of arriving COVID-19 patients allocated to hospital i on day t.

- $y_{i,t}$: COVID-19 patient occupancy in hospital i on day t.
- $o_{i,t}$: Total occupancy in hospital i on day t, defined as $o_{i,t} = d_{i,t} + y_{i,t}$.
- L_i: Random variable representing patient length of stay.
- $p(L_i \geq t)$: Probability that a patient at hospital i stays at least t days.
- β_i: Parameter for updating the estimated COVID-19 occupancy.
- s_t: State of the system at time t.

State: State $s_t \in \mathbb{R}^{2K+1}$ at time t consists of the COVID-19 and non-COVID-19 occupancies in each hospital, along with the number of arriving patients:

$$s_t = (y_{1,t}, y_{2,t}, \cdots, y_{K,t}; \quad d_{1,t}, d_{2,t}, \cdots, d_{K,t}; \quad N_t). \tag{82}$$

Action: Action $a_t \in \mathbb{R}^K$ at time t consists of the number of COVID-19 patients allocated to each hospital:

$$a_t = (a_{1,t}, a_{2,t}, \cdots, a_{K,t}) \tag{83}$$

subject to the constraints: $\sum_{i=1}^{N} a_{i,t} = N_t$ and $a_{i,t} \geq 0, \forall i = 1 \cdots K$. To allocate N_t patients to satisfy this constraint, we use an oracle function from Q-values, which receives N_t patients and Q-values, then outputs action a_t to K hospitals (Zuo and Joe-Wong, 2021; Li et al., 2024).

State Transition Function: The number of arriving COVID-19 patients is sampled from the corresponding stochastic process:

$$N_{t+1} \sim A_{t+1}. \tag{84}$$

The non-COVID-19 occupancy is sampled from the corresponding stochastic process (can be deterministic):

$$d_{i,t+1} \sim D_{i,t+1}. \tag{85}$$

The COVID-19 occupancy update:

$$y_{i,t+1} = \beta_i \cdot y_{i,t} + a_{i,t+1}, \tag{86}$$

where $\beta_i \in \mathbb{R}^K$ is provided by the environment, which can be done by minimizing the difference between the estimated and actual COVID-19 occupancies from the real-world dataset: COVID-19 Reported Patient Impact and Hospital Capacity by Facility, provided by the U.S. Department of Health & Human Services over $T = 1274$ days from 2020 to 2024. We use the $K = 40$ dataset collected from Texas hospitals.

Reward Function: The number of overflows across all hospitals:

$$r_t = \sum_{i=1}^{K} -\max(0, o_{i,t+1} - c_i), \tag{87}$$

where $o_{i,t+1} = y_{i,t+1} + d_{i,t+1}$ is the total occupancy of hospital i at time $t+1$. This represents the cost of a capacity shortage.

Baselines: We compare our method with: (1-2) combinatorial resource allocation UCB methods, i.e., UCB_RA and CUCB_RA (Zuo and Joe-Wong, 2021). And our Deep Q-learning extension, including (3) Deep Q-learning only (CNeural_RA, i.e., DQN) (Mnih et al., 2013; Paszke et al., 2019), and (4) Deep Q-learning with UCB exploration (Q-learning, i.e., DQN-UCB).

B.1.5 Source code and computing systems

Our source code includes the dataset scripts, setup for the environment, and our provided code (details in README.md). We run our code on a single GPU: NVIDIA RTX A6000-49140MiB with 8-CPUs: AMD Ryzen Threadripper 3960X 24-Core with 8GB RAM per each and require 10GB available disk space for storage.

B.2 Demo notebook code for Algorithm 1

```python
from rlberry.agents import IncrementalAgent, DiscreteCounter

class Ours(IncrementalAgent):
    def __init__(self):
        H = self.horizon
        S = self.env.observation_space.n
        A = self.env.action_space.n
        self.nu_kde = KernelDensity()
        self.de_kde = KernelDensity()
        self.list_nu_samples, self.list_de_samples = [], []
        # (s, a) visit counter
        self.N_sa = np.zeros((H, S, A))
        self.counter = DiscreteCounter(H, A)
        # Value functions
        self.V = np.ones((H+1, S))
        self.V[H, :] = 0
        self.Q = np.ones((H, S, A))
        self.Q_bar = np.ones((H, S, A))
        for hh in range(self.horizon):
            self.V[hh, :] *= (self.horizon-hh)
            self.Q[hh, :, :] *= (self.horizon-hh)
            self.Q_bar[hh, :, :] *= (self.horizon-hh)
        r_range = self.env.reward_range[1] - self.env.reward_range[0]
        self.v_max = np.zeros(self.horizon)
        for hh in reversed(range(self.horizon-1)):
            self.v_max[hh] = r_range + self.gamma*self.v_max[hh+1]

    def _get_action(self, state, hh=0):
        return self.Q_bar[hh, state, :].argmax()

    def _compute_bonus(self, n, hh, likelihood):
        bonus = self.bonus_scale_factor * np.sqrt(1.0 / n) + self.v_max[hh] / n
        bonus = min(bonus, self.v_max[hh])
        return bonus/likelihood

    def _update(self, state, action, next_state, reward, hh, likelihood):
        self.N_sa[hh, state, action] += 1
        nn = self.N_sa[hh, state, action]
        alpha = (self.horizon+1.0)/(self.horizon + nn)
        bonus = self._compute_bonus(nn, hh, likelihood)
        target = reward + bonus + self.gamma*self.V[hh+1, next_state]
        self.Q[hh, state, action] = (1-alpha)*self.Q[hh, state, action] + alpha * target
        self.V[hh, state] = min(self.v_max[hh], self.Q[hh, state, :].max())
        self.Q_bar[hh, state, action] = self.Q[hh, state, action]

    def _run_episode(self):
        # interact for H steps
        episode_rewards = 0
        state = self.env.reset()
        for hh in range(self.horizon):
            action = self._get_action(state, hh)
            next_state, reward, done, _ = self.env.step(action)
            episode_rewards += reward
            value1 = np.array([next_state, state, action])
            value2 = np.array([state, action])
            density_ratio = np.exp(self.nu_kde.score_samples(value1))
                /(np.exp(self.de_kde.score_samples(value2))
            self.counter.update(state, action)
            self._update(state, action, next_state, reward, hh, density_ratio)
            self.list_nu_samples.append(value1)
            self.list_de_samples.append(value2)
            self.nu_kde.fit(self.list_nu_samples[len(self.list_nu_samples)-100:])
            self.de_kde.fit(self.list_de_samples[len(self.list_nu_samples)-100:])
            state = next_state
            if done:
                break
        return episode_rewards
```

B.3 Additional results

B.3.1 Evaluation across different shift intensities

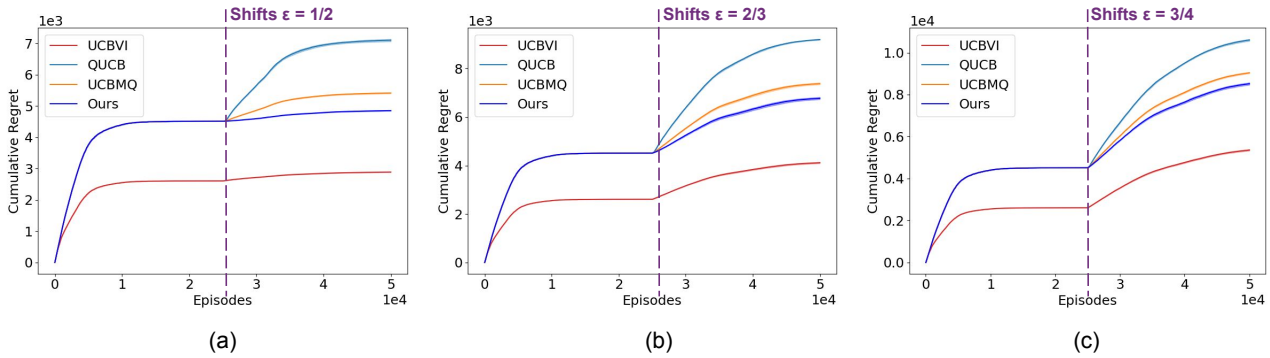


Figure 8: Cumulative regret comparison on Frozen-Lake with $K = 50000$ average over 10 runs, transition noise $\epsilon = 0$ before the shift $\bar{K} = 25000$. After that, $\epsilon = \{1/2, 2/3, 3/4\}$ in Figures (a), (b), and (c), respectively.

To evaluate the robustness of our method, we additionally compare it with other baselines across different shift intensities. Specifically, we deploy all models and test their performance with different levels of transition noise ϵ in the Frozen-Lake task. Figure 8 shows our result with $\epsilon = 0$ before the shift and $\epsilon = \{1/2, 2/3, 3/4\}$ after the shift in Figures 8 (a), (b), and (c), respectively. Firstly, we can see that our method consistently outperforms QUCB and UCBMQ by having a lower cumulative regret across different shift intensities. This once confirms our theoretical and empirical results in the main paper. Secondly, since the agent will move in the direction by following the action with probability $1 - \epsilon$ and move perpendicular to the intended direction with probability ϵ, we can see that when the transition noise ϵ increases, the performance of all methods will be degraded accordingly. Third, we observe that when the shift is too severe, such as $\epsilon = 3/4$ (i.e., moving randomly with probability 75%), all models will find it hard to converge, even with model-based RL like UCVI in Figure 8 (c).

B.3.2 Evaluation across different types of environment shifts

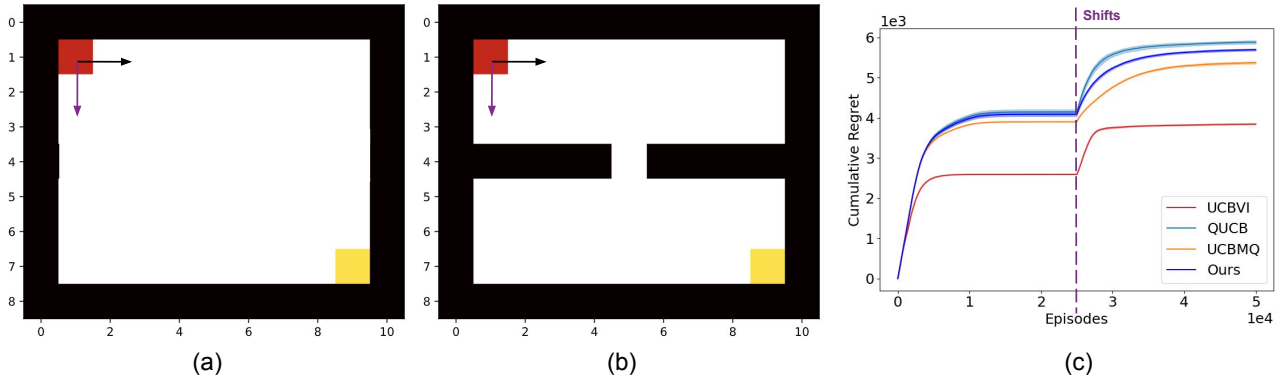


Figure 9: (a) The one-room GridWorld task: the starting state is shown in red, the rewarding state is shown in yellow, and the transitions are noisy with $\epsilon = 0.2$; (b) The environment shift of (a) by changing to two-rooms GridWorld map; (c) Cumulative regret comparison average over 10 runs, the map is one-room and two-rooms before and after the shift at $\bar{K} = 25000$, respectively.

Next, we consider a more real-world shift setting by changing the map of the GridWorld task. In particular, the agent starts to interact with the environment in the one-room map in Figure 9 (a) (similar to the environment before the shift in Figure 8). After that, the shift occurs in the episode $\bar{K} = 25000$, the environment changes to

a two-rooms map to interact with the agent in Figure 9 (b). Since the map suddenly changes, several optimal directions in one-room map from the starting point (red) to the ending point (yellow) will be blocked in two-rooms map. This leads to all methods suffering from low rewards (high regrets) between episodes 25000 and 30000 in Figure 9 (c). After that, they can adapt to the new map and converge with an optimal policy. Generally, we observe that the performance across the methods is quite similar to the previous results. Notably, the adaptation ability of our method is still better than QUCB, confirming the effectiveness of our UCB exploration in the non-stationary RL under distribution shifts.

B.3.3 Additional comparison with model-based RL

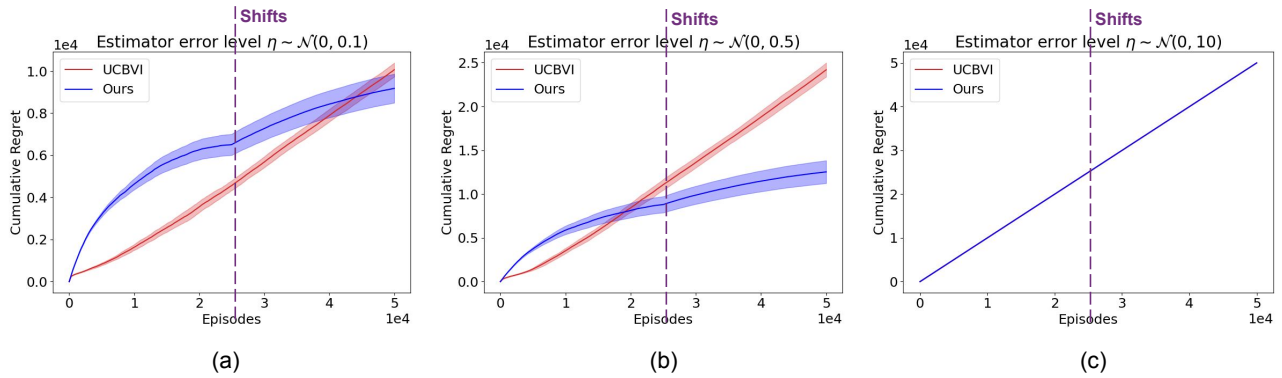


Figure 10: Cumulative regret comparison on Frozen-Lake with $K = 50000$ average over 10 runs, transition noise $\epsilon = 0$ and $\epsilon = 2/3$ before and after the shift at $\bar{K} = 25000$. The transition estimator error level $\eta \sim \mathcal{N}(0, std)$, where $std = \{0.1, 0.5, 10\}$ in Figures (a), (b), and (c), respectively.

Finally, we provide further ablation studies for comparison with model-based RL. Recall that our method does not explicitly model the transition operator $\mathbb{P}_h$ like the model-based method (e.g., UCBVI, UCBMQ, etc.), which needs to store and iterate through all possible $(s', s, a) \in \mathcal{S} \times \mathcal{S} \times \mathcal{A}$ tuples. On the one hand, this helps our method avoid a high computational burden, as discussed in Remark 3.1 and empirically shown in Figure 4. On the other hand, this helps our method avoid heavy dependence on the transition estimator, which may be inaccurate and lead to poor regret performance in model-based RL. Indeed, we provide an analysis on how the transition estimator error affects regret results between our method versus the model-based UCBVI as follows.

We inherit the experimental setting on Frozen-Lake in Figure 8 (b). In UCBVI, we denote $\hat{\mathbb{P}}_h$ as the transition estimator of the true underlying transition function $\mathbb{P}_h$. Hence, the transition estimator error is $||\hat{\mathbb{P}}_h - \mathbb{P}_h||$. To control the level of estimator error, we add a noise hypothesis η to the transition estimator, i.e., $\hat{\mathbb{P}}_h + \eta$ for every episode $h \in [H]$, where the r.v. $\eta \sim \mathcal{N}(0, std)$. Then, we use this transition estimator to run with both UCBVI and our algorithm. Since the transition estimator is the same, we can guarantee that the two methods have the same transition estimation error level.

We summarize the results in Figure 10, where different sub-figures represent different levels of estimator error by controlling the level of η. Firstly, we observe that the higher the level of randomness in η, i.e., the higher the transition estimator error, leading to worse performance for both methods. Moreover, if this randomness is too high, then both methods result in a linear regret in Figure 10 (c). Secondly, UCBVI is more sensitive to the transition estimator error than our method, causing UCBVI to become worse than our method when the transition estimator error increases. For example, when $\eta \sim \mathcal{N}(0, 0.1)$, UCBVI tends to be worse than us in the last round in Figure 10 (a). And when the error level of the estimator increases, that is, $\eta \sim \mathcal{N}(0, 0.5)$, we can clearly observe that UCBVI is worse than our method with a much higher cumulative regret in Figure 10 (b). Therefore, we can conclude that besides the computational limitation, model-based RL, such as UCBVI, is more sensitive to the transition estimator than our method. This leads to when the transition estimator error is high in some environments, our method could potentially bring out a better performance in both computational and regret (i.e., reward) performance than model-based RL.

B.3.4 Density estimation ablation studies

Method	$t = T/3$ (shift)	$t = 2T/3$ (shift)	$t = 3T/4$	$t = T$
QUCB	4512.0 ± 0.0	7571.0 ± 40.5	8129.0 ± 44.6	8951.0 ± 36.9
UCBVI	4513.5 ± 0.0	5737.25 ± 8.8	6101.0 ± 13.9	6703.25 ± 13.8
Ours (kernel)				
Linear	4515.5 ± 0.0	5139.25 ± 9.6	5441.0 ± 13.0	5970.0 ± 13.0
Cosine	4515.5 ± 0.0	5137.0 ± 8.6	5442.5 ± 12.9	5974.5 ± 13.0
Gaussian (default)	4513.5 ± 0.0	5126.0 ± 9.6	5425.0 ± 13.0	5954.0 ± 13.7
Exponential	4512.5 ± 0.0	5125.0 ± 8.8	5422.5 ± 12.5	5948.0 ± 13.5
Ours (bandwidth)				
0.01	4513.5 ± 0.0	5598.0 ± 10.1	6055.25 ± 16.5	6501.5 ± 20.7
0.1	4513.0 ± 0.0	5123.5 ± 8.8	5420.0 ± 12.6	5947.5 ± 13.7
1.0 (default)	4513.5 ± 0.0	5126.0 ± 9.6	5425.0 ± 13.0	5954.0 ± 13.7
10	4513.5 ± 0.0	7526.0 ± 38.9	8121.25 ± 37.3	8949.0 ± 33.0
Ours (window sizes)				
1	4513.0 ± 0.0	7514.25 ± 36.7	8122.0 ± 44.0	8943.0 ± 36.5
10	4513.0 ± 0.0	6098.25 ± 10.1	6391.0 ± 16.6	7007.0 ± 21.4
100 (default)	4513.5 ± 0.0	5126.0 ± 9.6	5425.0 ± 13.0	5954.0 ± 13.7
1000	4513.5 ± 0.0	5049.0 ± 9.0	5311.25 ± 12.5	5729.25 ± 12.8

Table 2: Ablation studies in cumulative regret (lower is better) across different density function's parameters, including kernel type, bandwidth, and window size. Results are reported over 10 runs. Experiments correspond to our Fig 3 (b) on Frozen-Lake with two shifts at time $t = T/3$ and $t = 2T/3$, where $T = 60000$.

Table 2 utilizes different kernel density function parameters, including various kernel types, different bandwidth sizes, and different window sizes, in the Frozen-Lake environment. We observe that our method is quite robust across different kernel types (e.g, Gaussian, Exponential, Cosine, and Linear) and the bandwidth between the range of (0.01, 1.0). If the bandwidth is too large (e.g., 10), then the KDE is oversmoothed and biased, leading to a degradation in regret performance and resulting in a similar result with QUCB. Similarly, if the window size is too small (e.g., (1,10)), the density function will also be biased by only having a small observation. That said, with a sufficient number of samples (100,1000), our method consistently outperforms QUCB and UCBMQ.

B.3.5 Extension to continuous action space on MuJoCo's locomotion

Method on Hopper	$t = T/2$ (shift)	$t = 2T/3$ (shift)	$t = 3T/4$	$t = T$
DQN	179.4 ± 1.0	221.0 ± 1.4	295.3 ± 1.5	410.2 ± 2.3
DQN-UCB	211.0 ± 0.8	257.2 ± 1.2	331.1 ± 1.3	509.0 ± 1.7
Ours	211.8 ± 0.8	397.5 ± 1.2	516.4 ± 1.2	689.6 ± 1.4
Method on Halfcheetah				
DQN	1,410.1 ± 2.4	1,595.6 ± 2.6	1,800.8 ± 3.0	1,967.2 ± 3.8
DQN-UCB	1,526.1 ± 2.0	1,733.4 ± 2.3	1,920.7 ± 2.6	2,125.0 ± 2.5
Ours	1,526.7 ± 2.0	2,017.6 ± 2.2	2,293.6 ± 2.3	2,694.9 ± 2.2
Method on Walker				
DQN	288.5 ± 0.9	319.1 ± 1.0	380.7 ± 1.2	477 ± 1.4
DQN-UCB	323.3 ± 0.7	367.0 ± 0.7	433.3 ± 0.8	595 ± 0.8
Ours	323.8 ± 0.6	411.6 ± 0.6	631.6 ± 0.6	755 ± 0.7

Table 3: Experimental results in average return (higher is better) on MuJoCo's locomotion, including hopper, halfcheetah, and walker environments across $T = 200000$. Results are reported over 10 runs.

Although we focus on Q-learning (which is designed for discrete action spaces), our framework can also be extended for continuous action spaces by using the Discretizing Continuous Action Space method (Tang and Agrawal, 2020). Specifically, given action space $\mathcal{A} = [-1, 1]^m$ in Mojuco Locomotion, we discretize each dimension of the action space into $K = 100$ equally spaced atomic actions (bins). The set of atomic actions for any dimension i is $\mathcal{A} = [\frac{2j}{K-1} - 1]_{j=0}^{K-1}$. During inference time, we convert the discrete Deep-Q-Learning outputs to continuous w.r.t. the mean of each bin. Table 3 shows that our method also outperforms other baselines, achieving a higher average return over 20 evaluation episodes across MuJoCo Locomotion's tasks. This result is consistent with the lowest cumulative regret, confirming the effectiveness of our method's shift awareness under distribution shifts.