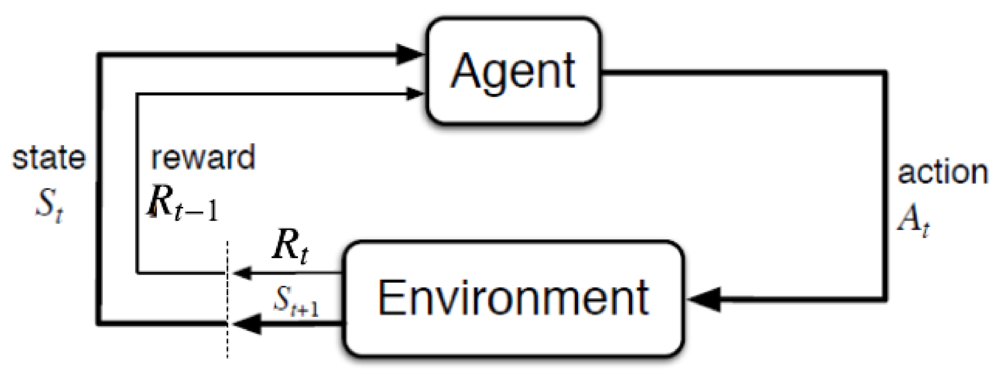
How Does an Approximate Model Help in Reinforcement Learning?

Fei Feng, Wotao Yin, and Lin Yang

fei.feng@math.ucla.edu; wotaoyin@math.ucla.edu; linyang@ee.ucla.edu

Background

Markov Decision Process (MDP)



$\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, R, \gamma)$.

- $t = 0, 1, 2, \ldots$
- $S_t \in \mathcal{S}$, state space. $A_t \in \mathcal{A}$, action space.
- $A_t \leftarrow \pi(S_t)$, π is a policy
- $R_t(S_t, A_t) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, reward.
- $P(S_{t+1}|S_t, A_t)$, transition probability.

Reinforcement Learning (RL)

- The mathematical model of RL is MDP but no knowledge of P and R.
- The target is to learn an optimal policy:

$$\underset{\pi}{\text{maximize}}\ \mathbb{E}^\pi\left[\sum_{t=0}^{\infty} \gamma^t R_t(S_t, A_t)\right],$$

  with transition samples (S_t, A_t, S_{t+1}, R_t).
- In real applications, it takes a lot of time and samples to learn. **How to learn faster with fewer samples is a central challenge in RL.**
- One approach is transfer learning, i.e. borrow knowledge from previously-learned **similar** tasks to help learn new ones.

Fundamental Questions

1. How to define *similarity* between models?
2. How much benefit can we gain from an approximate model?

What we consider?

1. A natural candidate to measure similarity is probabilistic distance. Given two MDPs $\mathcal{M}_0 := (\mathcal{S}, \mathcal{A}, p_0, r_0, \gamma)$ and $\mathcal{M} := (\mathcal{S}, \mathcal{A}, p, r, \gamma)$, we define

$$d_{\mathrm{TV}}(\mathcal{M}_0, \mathcal{M}) := \max\Big(\max_{(s,a)\in\mathcal{S}\times\mathcal{A}} \|p_0(\cdot|s,a) - p(\cdot|s,a)\|_1, \|r_0 - r\|_\infty\Big).$$

2. Given the full knowledge of $\mathcal{M}_0$ such that $d_{\mathrm{TV}}(\mathcal{M}_0, \mathcal{M}) \le \beta$, how many samples do we need to learn a near-optimal policy for $\mathcal{M}$?

High-level Idea

High-level Idea

- Based on the knowledge of $\mathcal{M}$, we can estimate the value of each action in $\mathcal{M}'$.
- Based on the interval estimation, we define two types of actions:
  - **potentially-optimal action**: action that has a chance to be optimal in $\mathcal{M}'$;
  - **must-learn action**: action that if we do not learn, we can fail to construct a near-optimal policy in $\mathcal{M}'$.
- Establish sample complexity results using the number of potentially-optimal actions and the number of must-learn actions.

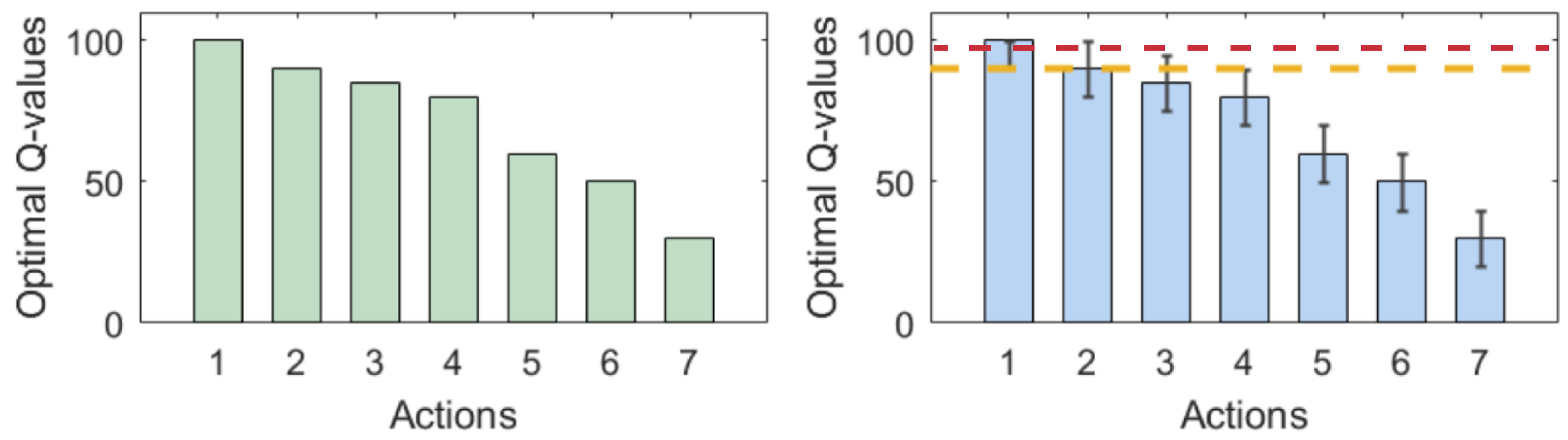


Figure 2: An illustration to the idea. The left graph depicts values in $\mathcal{M}$, which provides interval estimate of values in $\mathcal{M}'$ as in the right graph (the range bars on top of the columns). An action is potentially-optimal if the top point of its range bar can exceed the yellow dash line; it is must-learn if the top point is above the red dash line. The value of the lines depends on $\mathcal{M}$ and β.

Main Results

Theorem 1 *(Upper Bound)* *For any $\varepsilon, \delta \in (0,1)$, the sample complexity of learning an ε-optimal policy for $\mathcal{M}'$ with probability at least $1-\delta$ is*

$$\tilde{O}\left(\frac{\overline{N}(\mathcal{M}, \beta, \varepsilon)}{(1-\gamma)^3\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right),$$

where $\overline{N}$ is the number of potentially-optimal actions in $\mathcal{M}'$.

Theorem 2 *(Lower bound) There exists $\varepsilon_0, \delta_0 \in (0,1)$ such that for all $\varepsilon \in (0, \varepsilon_0)$, $\delta \in (0, \delta_0)$, the sample complexity of learning an ε-optimal policy for $\mathcal{M}'$ with probability at least $1-\delta$ is*

$$\Omega\left(\frac{\underline{N}(\mathcal{M}, \beta, \varepsilon)}{(1-\gamma)^3\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right),$$

where $\underline{N}$ is the number of must-learn actions in $\mathcal{M}'$.

Best Case: If $\overline{N} = 1$, the optimal policy of $\mathcal{M}$ is also an optimal policy of $\mathcal{M}'$. Policy is transferable.

Worst Case: If $\underline{N}$ is $\Omega(|S||A|)$, the approximate model is basically of no-use.

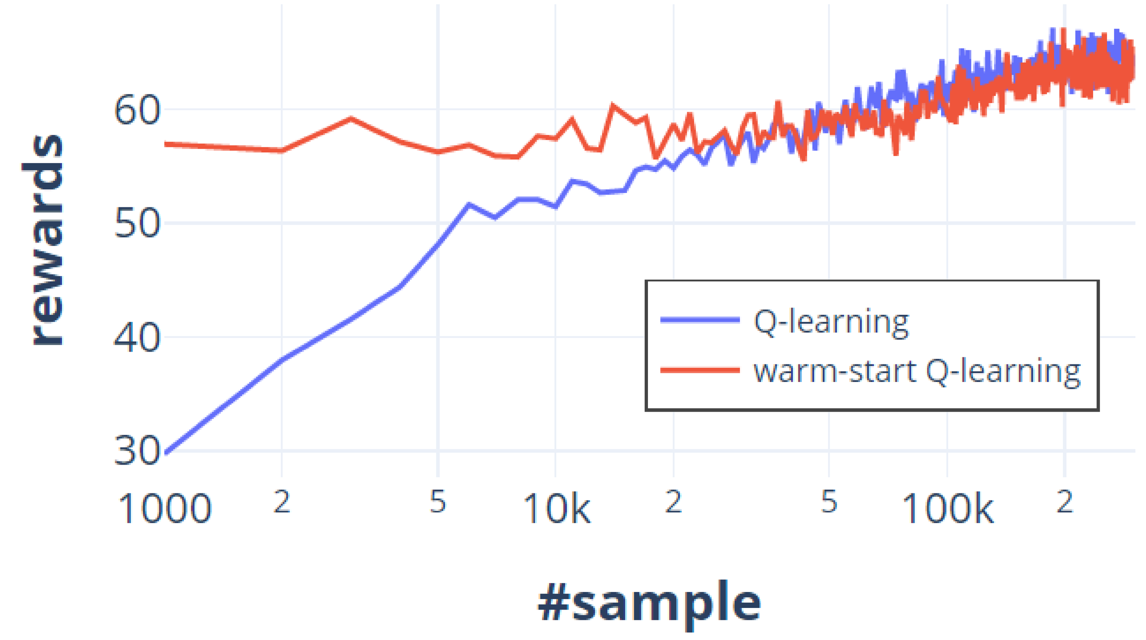


Figure 1: An illustration to the worst case. The blue line is learning-from-scratch, the red line is learning with the knowledge of an approximate model. The warm-start algorithm achieves a jump-start performance but asymptotically, the numbers of samples required with or without an approximate model are of the same level.