Upper Confidence Bounds Action-values

Martha White
Assistant Professor
University of Alberta
Thanks to collaborators Raksha Kumaraswamy, Matthew Schlegel, Adam White, Sungsu Lim

amii

UNIVERSITY OF ALBERTA

Goal for this Talk

- Discuss one direction for using upper confidence bounds on action-values in reinforcement learning
- Highlight some open questions and issues

Problem Setting

- General state and action space (finite, continuous)
- Agent estimates action-values, from stream of interaction
  - $Q^*(s,a)$ = expected return under optimal policy
- How can the agent be confident in its estimates of $Q^*(s,a)$?
- **Our goal**: directed exploration to efficiently estimate $Q^*(s,a)$

Many model-free methods use Uncertainty Estimates

My Preference. But how do we do it? ↙

- Option 1: Estimate uncertainty in Q(s,a)
  - Use upper confidence bounds or Thompson sampling
- Option 2: Reward Bonuses
  - Add reward upper bound to the reward in the Q-learning update

Upper Confidence Bounds for Stochastic Bandits

- No states: Estimate action-values Q(a)
  - Q(a) = expected reward for taking that action
- Estimating a sample mean, so can estimate confidence interval around that estimate
  - e.g., if true Q(a) normally distributed, use U(a) = 1.96 sqrt(var(Q(a)) / t)
  - e.g., unknown distribution, use concentration inequality (e.g., Hoeffding)
- Select action with highest plausible value (optimism!)

$$\arg\max_a \hat{Q}(a) + \hat{U}(a)$$

Upper Confidence Bounds for (iid) Contextual Bandits

- Q(s,a) = expected reward for taking that action, in s
- Estimating a conditional sample mean; can use methods from regression to estimate confidence
  - e.g., if Q(s,a) is a linear function of features x(s,a), Q(s,a) = < x(s,a), w> can use known formula for variance of weights in linear regression
  - matrix C = variance of weights, reflects that w would have been different had other streams of data been observed

Upper Confidence Bounds for (iid) Contextual Bandits

- Q(s,a) = expected reward for taking that action, in s

- Estimating a conditional sample mean; can use methods from regression to estimate confidence

$$\text{With probability } 1-p,$$

$$\mathbf{x}^\top \mathbf{w}^* \leq \mathbf{x}^\top \mathbf{w}_t + \sqrt{\frac{p+1}{p}}\sqrt{\mathbf{x}^\top \mathbf{C}\mathbf{x}}$$

$$\hat{U}(s,a) = \sqrt{\frac{p+1}{p}}\sqrt{\mathbf{x}^\top \mathbf{C}\mathbf{x}}$$

- Select action with highest plausible value (optimism!)

$$\arg\max_a \hat{Q}(s,a) + \hat{U}(s,a)$$

Why is RL different from the contextual bandit setting?

- **Temporal connections:** Actions now influence the context (states) and cumulative rewards into the future
- **Bootstrapping:** Do not get a sample of the target (return), particularly since policy is changing

UCB for Policy Evaluation

- **For a fixed policy**, we can obtain UCB on action-values
- Previous algorithms used supervised learning approaches (e.g., Bayesian linear regression) to estimate distribution over weights (some of these could be used to get UCB)
    - approximation assumes bootstrapped target does not include weights, such as by using a target network
- We have a sound UCB designed for an RL algorithm, under linear value function approximation

Key idea to get UCB for a fixed policy

- We can characterize the Variance(w), because we use a particular update on w (LSTD)
- Still Cheybshev's inequality, but now C is different

$$\text{With probability } 1 - p,$$
$$\mathbf{x}^\top \mathbf{w}^* \leq \mathbf{x}^\top \mathbf{w}_t + \sqrt{\tfrac{p+1}{p}}\sqrt{\mathbf{x}^\top \mathbf{C} \mathbf{x}}$$

Extension to control

- Current methods act greedily according to uncertainty estimates for the **current** policy (or set of policies)
  - Many approaches implicitly do this, including RLSVI, UCBootstrap, Bayesian Deep Q Networks, Bootstrap DQN, UCLS (ours)
- Ensure variance estimates start large enough,
  - to reflect lack of certainty in the variance estimates themselves
  - to ensure true model possible under prior

High-level approach

- Iteratively update action-values (policy) and upper confidence estimates
  - a generalized policy iteration strategy, but plus uncertainty
- Overestimate (or initialize large) the variance of the weights
- …But does this work?

What does the theory say?

- RLSVI has an optimality result (tabular, finite horizon)
- The key concept is **stochastic optimism**

$$\text{For some } T > 0, \text{ for every } t \geq T \\ \text{with } \tilde{Q}_t(s,a) = \hat{Q}_t(s,a) + \hat{u}_t(s,a)$$

$$\mathbb{E}[\tilde{Q}_t(S,A)] \geq \mathbb{E}[Q^*(S,A)]$$

High-level result

If we have stochastic optimism,
shrinking confidence interval radius with rate $f(t)$ and
action-value update where $\hat{Q}_t$ approaches Q^{π_t} with rate $g(t)$,
then the regret is

$$\mathbb{E}[Q^*(S,A)] - \mathbb{E}[Q^{\pi_t}(S,A)] \leq f(t) + g(t)$$

where π_t acts greedily according to $\tilde{Q}_t = \hat{Q}_t + \hat{u}_t$

…But this does not seem to match the algorithm design

- The algorithms iterate confidence intervals with the current policies
- Stochastic optimism is about Q*
- Yet the algorithms seem to work well in practice, and RLSVI was proven to converge

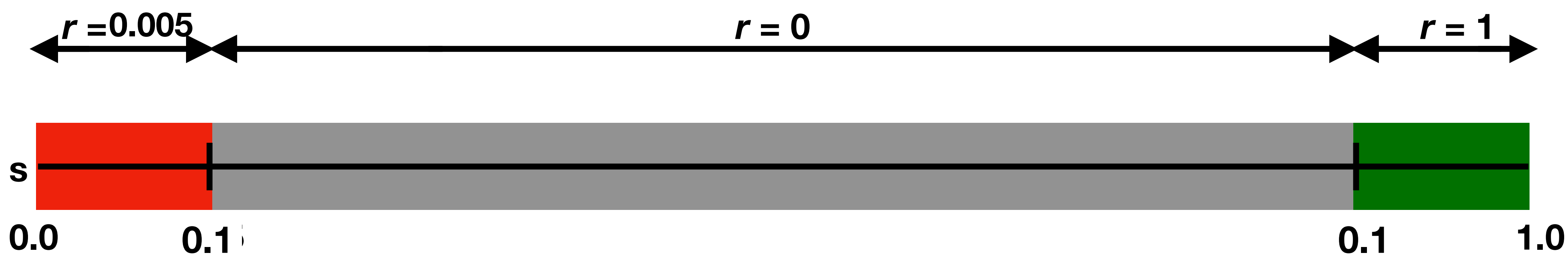
Convergence for RLSVI

- Finite-horizon setting enabled estimates to become correct for myopic one-step rewards
- Variance estimates needed to be initialized sufficiently high
- Tabular: no issues with generalization causing the variance in a state to be incorrectly reduced, before it is visited

Empirically algorithms with this flavour seem to work well

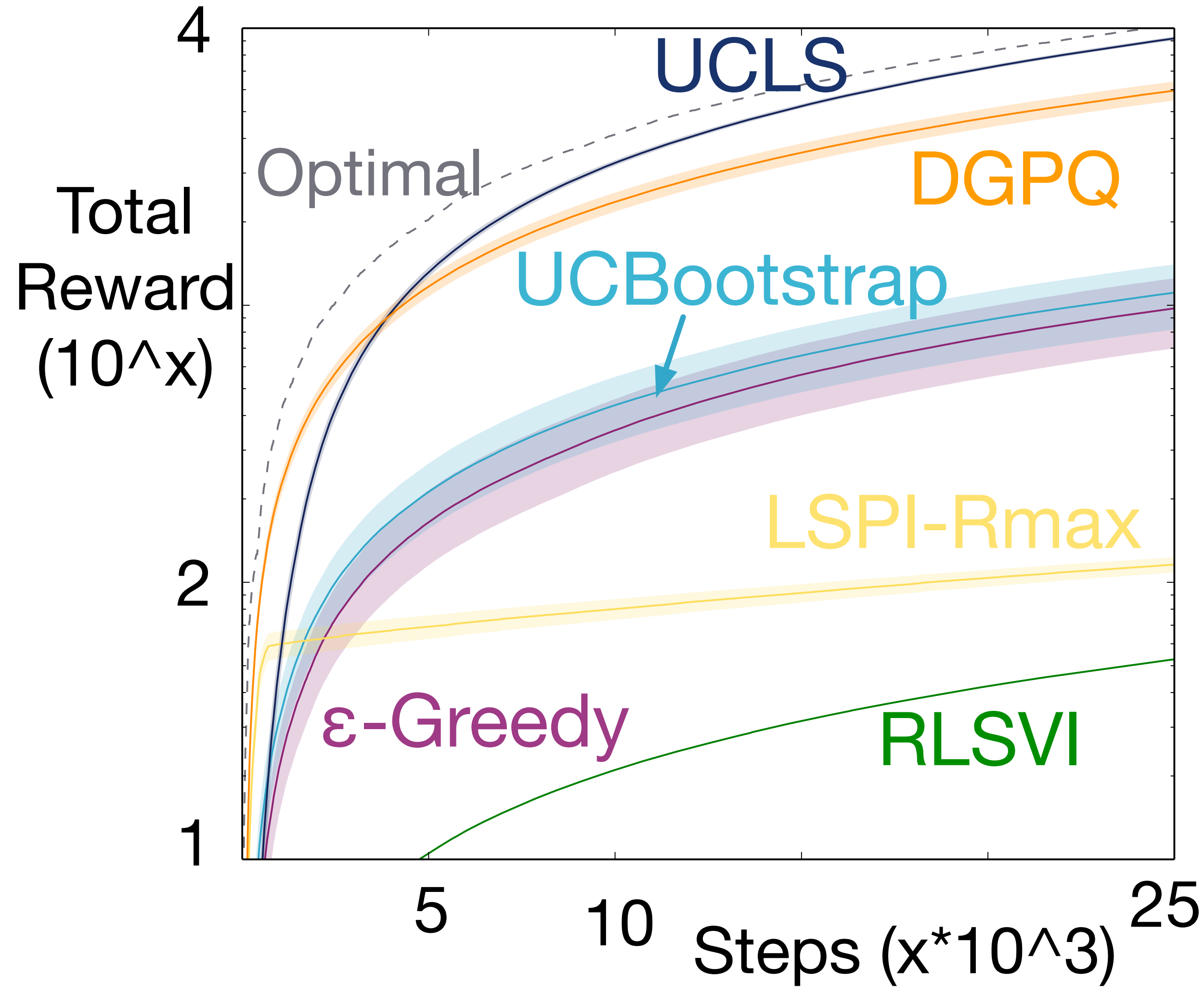
- Bootstrap DQN
- Bayesian Deep Q Networks
- Double Uncertain Value Networks
- UCLS (our algorithm)

Experiments in RiverSwim



- Continuing problem with gamma = 0.99
- Stochastic displacement with 0.1
- Starts near the left (random start location)
- 100 runs, with our parameters fixed across all domains

UCLS learns effectively



Riverswim

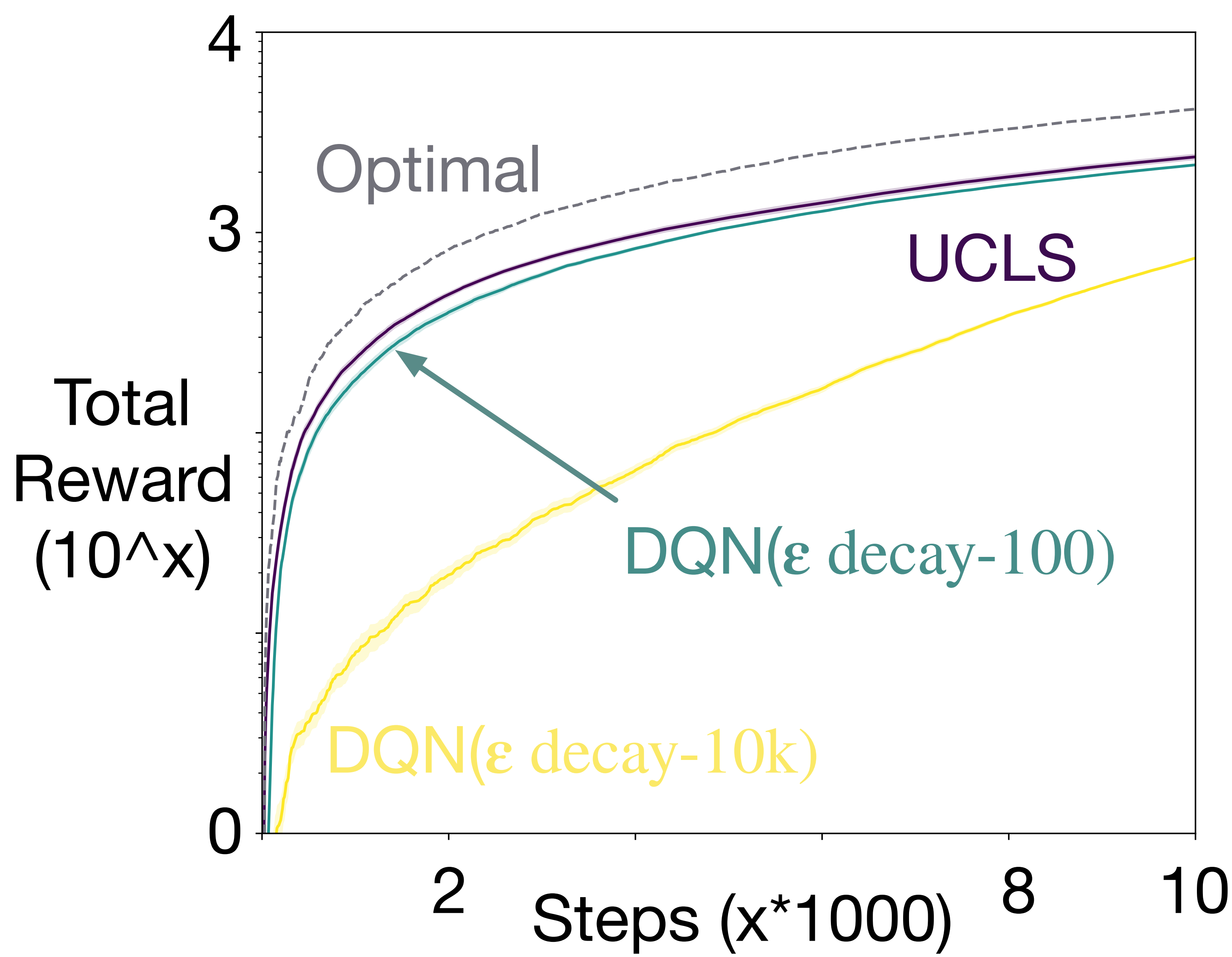
UCBootstrap fails, but does not ensure variance is sufficiently large

RLSVI learns slowly, likely because needs to simulate entire episodes

…Even works outside derived settings

- A natural idea: using algorithms designed for a linear setting and apply them to the last layer of an NN
- UCLS and Bayesian Deep Q Networks both do this

UCLS+NNs



Nice that we can derive confidence bounds for the linear setting (more feasible), and still benefit from them when learning the representation

Open Questions: Estimating UCB for control

- Do we have to estimate UCB directly on Q*?
- Can we estimate UCB on Q^{π}, and iterate?
  - Is it possible to prove convergence to optimal, for these algorithms that do seem to perform well in practice?
- Is it useful to use UCB derived for fixed policies, but with inflated estimates of variance, to get stochastic optimism?