Multi-Step Reinforcement Learning: A Unifying Algorithm

group:王軒081605、朱以箴M093567、謝承恩0816183、莊至浩0816056

Introduction

- Unifying seemingly disparate algorithmic ideas to produce better performing algorithms has been a longstanding goal in reinforcement learning.
- Currently, there are a multitude of algorithms that can be used to perform TD control, including Sarsa, Q-learning, and Expected Sarsa.
- This paper proposed a new multi-step action-value algorithm called Q(σ) that unifies these algorithms by introducing a new parameter σ.
- σ controls the degree of sampling performed by the algorithm at each time-step, which can be continuously varied.

TD Algorithms

- TD(0)
- TD(λ)
- Sarsa
- Expected Sarsa
- Q-learning

Atomic Multi-Step Algorithms

- The TD methods presented in the previous section can be generalized even further by bootstrapping over longer time intervals. This can decrease the bias of the update at the cost of higher variance.
- So, the number of steps to take per update to optimize the performance becomes another important problem to solve.

Tree-Backup

Instead of taking expectation over the actions at the last step of the back up like in Expected SARSA, Tree-Backup take expectation over actions at every step, and just like Expected SARSA and Q-learning, this method does not require importance sampling to apply off-policy.

The Q(σ) Algorithm

Concept

All of the algorithms presented so far can be broadly categorized into two families:

(1) Those that backup their actions as samples.
(2) Those that consider an expectation over all actions in their backup.

Q(σ) unifies both family together.

- The idea is to do back up by (1), or by (2), but the choice doesn't have to be strictly either side.
- σ, which controls the weighting between (1) and (2) doesn't have to remain constant for every step of the back up.

The Algorithm

This is the pseudocode for the complete off-policy n-step Q(σ) algorithm.

TD error is the weighted average of SARSA and expected SARSA:

$$\begin{aligned}\delta_t^\sigma &= \sigma_{t+1}\delta_t^S + (1-\sigma_{t+1})\delta_t^{ES},\\ &= R_{t+1} + \gamma[\sigma_{t+1}Q_t(S_{t+1}, A_{t+1}) + (1-\sigma_{t+1})V_{t+1}]\\ &\quad - Q_{t-1}(S_t, A_t).\end{aligned} \tag{13}$$

The n-step return is then:

$$\begin{aligned}G_{t:t+n} &= Q_{t-1}(S_t, A_t)\\ &\quad + \sum_{k=t}^{\tau} \delta_k^\sigma \prod_{i=t+1}^{k} \gamma[(1-\sigma_i)\pi(A_i|S_i) + \sigma_i].\end{aligned} \tag{14}$$

Algorithm 1 Off-policy n-step $Q(\sigma)$ for estimating q_π

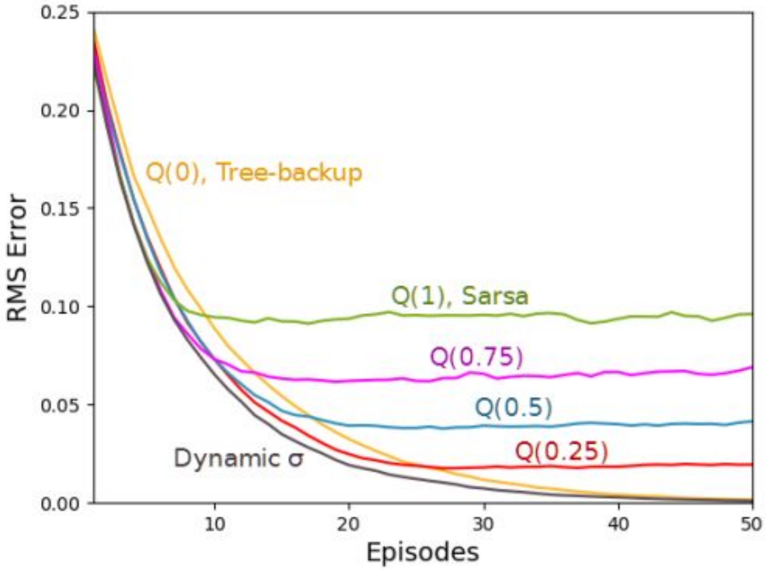
Input: a behaviour policy μ and a target policy π
Initialize $S_0 \neq$ terminal; select A_0 according to $\mu(.|S_0)$
Store S_0, A_0, and $Q(S_0, A_0)$
for $t = 0, 1, 2, ..., T+n-1$ **do**
  if $t < T$ **then**
    Take action A_t; observe R and S_{t+1}
    Store S_{t+1}
    if S_{t+1} is terminal **then**
      Store: $\delta_t^\sigma = R - Q(S_t, A_t)$
    else
      Select A_{t+1} according to $\mu(\cdot|S_{t+1})$ and Store
      Store: $Q(S_{t+1}, A_{t+1})$, σ_{t+1}, $\pi(A_{t+1}|S_{t+1})$
      Store: $\delta_t^\sigma = R + \gamma[\sigma_{t+1}Q(S_{t+1}, A_{t+1})$
        $+(1-\sigma_{t+1})V_{t+1}] - Q(S_t, A_t)$
      Store: $\rho_{t+1} = \frac{\pi(A_{t+1}|S_{t+1})}{\mu(S_{t+1}|A_{t+1})}$
    end if
  end if
  $\tau \leftarrow t - n + 1$
  if $\tau \geq 0$ **then**
    $\rho \leftarrow 1$
    $E \leftarrow 1$
    $G \leftarrow Q(S_\tau, A_\tau)$
    for $k = \tau, ..., \min(\tau+n-1, T-1)$ **do**
      $G \leftarrow G + E\delta_k^\sigma$
      $E \leftarrow \gamma E\left[(1-\sigma_k)\pi(A_{k+1}|S_{k+1}) + \sigma_{k+1}\right]$
      $\rho \leftarrow \rho(1 - \sigma_k + \sigma_k\rho_k)$
    end for
    $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\rho[G - Q(S_\tau, A_\tau)]$
  end if
end for

Experiment

19-State Random Walk

This is the experiment result on the simple 19-State Random Walk.

The results are an average of 100 runs, and the standard errors are all less than 0.006. Q(1) had the best initial performance, Q(0) had the best asymptotic performance, and dynamic σ outperformed all fixed values of σ.



Mountain Cliff

The first graph's results are for selected α values, then are connected by straight lines, and are an average of 1000 runs. The standard errors are all less than 0.3 which is about a line width. 3-step algorithms performed better than their 1-step equivalents, and Q(σ) with a dynamic σ performed the best overall.

The dark lines show the results smoothed using a right-centered moving average with a window of 30 successive episodes, while the light lines show the un-smoothed results. Q(σ) with dynamic σ had the best performance among all the algorithms.