

AIL 722: Reinforcement Learning

Lecture 21: Midterm Review

Raunak Bhattacharyya



ScAI

YARDI SCHOOL OF ARTIFICIAL INTELLIGENCE
INDIAN INSTITUTE OF TECHNOLOGY DELHI

Question

- Suppose MDP with 100 states and 4 actions (up, down, left, right)
- Policy improvement step:

$$\pi^{(k+1)}(s) = \arg \max_a \left[r(s, a) + \gamma \sum_{s'} T(s' | s, a) V^{\pi^{(k)}}(s') \right]$$

- Assume if there are ties between actions, they are broken in order.

Is it possible that $\pi^{(2)} \neq \pi^{(3)}$ but $\pi^{(2)} = \pi^{(4)}$?

Question

- Suppose MDP with discrete state space (size n) and action space (size m)
- What is the time complexity of the policy improvement step?
- If we know transition probs belong to $\{0,1\}$ can we give a tighter complexity bound?

Question

- Suppose MDP with 10 possible states and 5 possible actions
- Suppose every state-action pair has a non-zero probability of transitioning to every state
- For each iteration of VI, where a single iteration corresponds to all the states being updated, how many times will we need to evaluate the transition function?

Question

- MDP with 5 states ($s^{1:5}$) and 2 actions: stay and continue. We know that:

$$T(s_i | s_i, a_S) = 1 \text{ for } i \in \{1, 2, 3, 4\}$$

$$T(s_{i+1} | s_i, a_C) = 1 \text{ for } i \in \{1, 2, 3, 4\}$$

$$T(s_5 | s_5, a) = 1 \text{ for all actions } a$$

$$R(s_i, a) = 0 \text{ for } i \in \{1, 2, 3, 5\} \text{ and for all actions } a$$

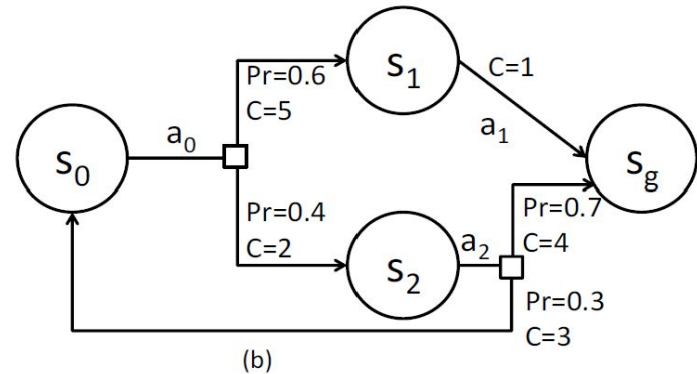
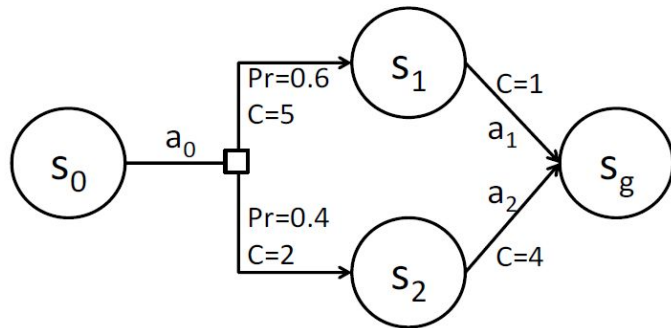
$$R(s_4, a_S) = 0$$

$$R(s_4, a_C) = 10$$

What is the discount factor γ if the optimal value $V^*(s^1) = 1$?

Question

- Compute the value function for the following:



Question

- What is the complexity of Q-learning if we interact with the environment for d steps?

- What is the complexity of SARSA?

Midterm Review

Conditional Expectations

$$J(\theta) = \mathbb{E}_{(s_1, a_1, s_2, a_2, \dots, s_T, a_T) \sim p_\theta(s_1, a_1, \dots, s_T, a_T)} \left[\sum_{t=1}^T r(s_t, a_t) \right]$$

$$J(\theta) = \mathbb{E}_{s_1 \sim p(s_1)} \left[\mathbb{E}_{a_1 \sim \pi_\theta(a_1 | s_1)} \left[r(s_1, a_1) + \mathbb{E}_{s_2 \sim p(s_2 | s_1, a_1)} \left[\mathbb{E}_{a_2 \sim \pi_\theta(a_2 | s_2)} \left[r(s_2, a_2) + \dots \mid s_2 \right] \mid s_1, a_1 \right] \mid s_1 \right] \right]$$

Introducing the Q-function

$$J(\theta) = \mathbb{E}_{s_1 \sim p(s_1)} \left[\mathbb{E}_{a_1 \sim \pi_\theta(a_1 | s_1)} \left[r(s_1, a_1) + \mathbb{E}_{s_2 \sim p(s_2 | s_1, a_1)} \left[\mathbb{E}_{a_2 \sim \pi_\theta(a_2 | s_2)} \left[r(s_2, a_2) + \dots \mid s_2 \right] \mid s_1, a_1 \right] \mid s_1 \right] \right]$$

Suppose we knew this part

Definition: Q-function

$$Q^\pi(s_t, a_t) = \mathbb{E} \left[\sum_{t'=t}^T r(s_{t'}, a_{t'}) \mid s_t, a_t \right]$$

$$J(\theta) = \mathbb{E}_{s_1 \sim p(s_1)} \left[\mathbb{E}_{a_1 \sim \pi_\theta(a_1 | s_1)} \left[Q(s_1, a_1) \mid s_1 \right] \right]$$

Definition: Value Function (V)

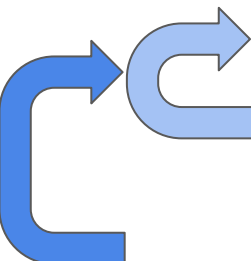
$$J(\theta) = \mathbb{E}_{s_1 \sim p(s_1)} \left[\mathbb{E}_{a_1 \sim \pi_\theta(a_1 | s_1)} \left[Q(s_1, a_1) \mid s_1 \right] \right]$$

$$V^\pi(s_t) = \mathbb{E}_{a_t \sim \pi_\theta(a_t | s_t)} \left[Q^\pi(s_t, a_t) \mid s_t \right]$$

$$V^\pi(s_t) = \mathbb{E} \left[\sum_{t'=t}^T r(s_{t'}, a_{t'}) \mid s_t \right]$$

$$J(\theta) = \mathbb{E}_{s_1 \sim p(s_1)} \left[V^\pi(s_1) \right]$$


Policy Iteration: Using Q-values

- 
1. $V^\pi(s) = r(s, \pi(s)) + \gamma \cdot \mathbb{E}_{p(s'|s, \pi(s))} [V^\pi(s')]$
 2. Set $\pi \leftarrow \pi_{\text{new}}$

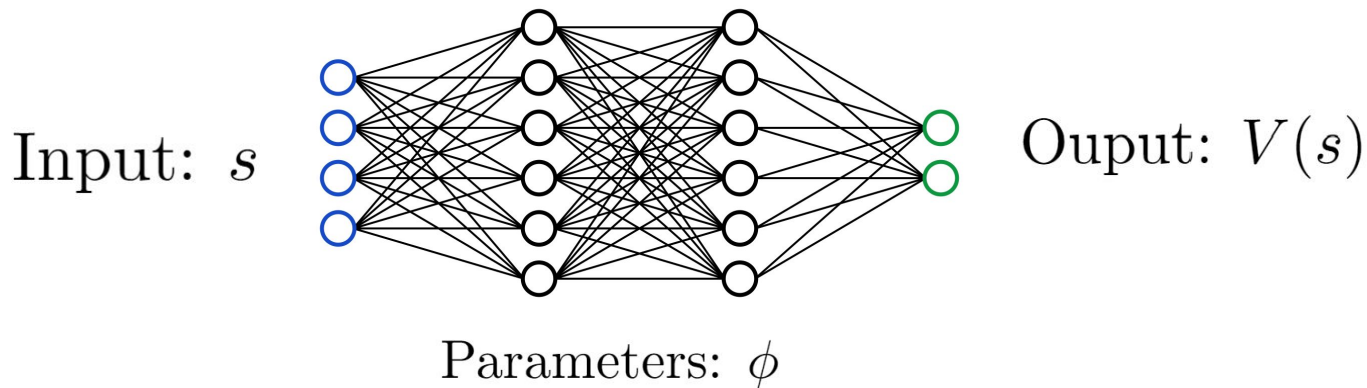
$$\pi_{\text{new}} = \begin{cases} 1 & \text{if } a = \arg \max_a Q^\pi(s, a) \\ 0 & \text{otherwise} \end{cases}$$

Value Iteration

Start with a random value function $V(s)$

- 
1. Set $Q(s, a) \leftarrow r(s, a) + \gamma \cdot \mathbb{E}_{p(s'|s,a)} \left[V^\pi(s') \right]$
 2. Set $V(s) \leftarrow \max_a Q(s, a)$

Loss Function



$$L(\phi) = \frac{1}{2} \left\| V_{\phi}(s) - \max_a Q^{\pi}(s, a) \right\|^2$$

Fitted Value Iteration

$$\text{Dataset: } \left\{ (s_i, a_i, s'_i, r_i) \right\}$$

1. Set $y_i \leftarrow \max_a \left(r(s_i, a_i) + \gamma \cdot \mathbb{E} \left[V_\phi(s'_i) \right] \right)$
2. Set $\phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|V_\phi(s_i) - y_i\|^2$

Fitted VI: Restrictive Assumption

1. Set $y_i \leftarrow \max_a \left(r(s_i, a_i) + \gamma \cdot \mathbb{E} \left[V_\phi(s'_i) \right] \right)$

2. Set $\phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|V_\phi(s_i) - y_i\|^2$

There are two places where we require knowledge of the transition dynamics

Compute expected value at next state

Take max over actions (needs us to be able to try out all possible actions from the same state)

Need an MDP simulator: to try out every action, get next state and reward

Does not match up to experience-based learning in general. Cannot go back to exact same state to try out new actions.

Fitted Value Iteration

1. Set $y_i \leftarrow \max_{a_i} \left(r(s_i, a_i) + \gamma \cdot \mathbb{E} \left[V_\phi(s'_i) \right] \right)$

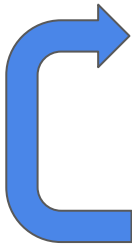
2. Set $\phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|V_\phi(s_i) - y_i\|^2$

We don't have a simulator. We only have a start state. We can sample trajectories

Underlying Idea: VI

1. Set $y_i \leftarrow \max_a \left(r(s_i, a_i) + \gamma \cdot \mathbb{E} \left[V_\phi(s'_i) \right] \right)$

2. Set $\phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|V_\phi(s_i) - y_i\|^2$



1. Set $Q(s, a) \leftarrow r(s, a) + \gamma \cdot \mathbb{E}_{p(s'|s,a)} \left[V^\pi(s') \right]$
2. Set $V(s) \leftarrow \max_a Q(s, a)$

How do we get to a model-free algorithm?

Fitted Q-Iteration

1. Set $Q(s, a) \leftarrow r(s, a) + \gamma \cdot \mathbb{E}_{p(s'|s, a)} \left[V^\pi(s') \right]$
2. Set $V(s) \leftarrow \max_a Q(s, a)$

$$\text{Set } V(s) \leftarrow \max_a Q(s, a)$$

The crux element

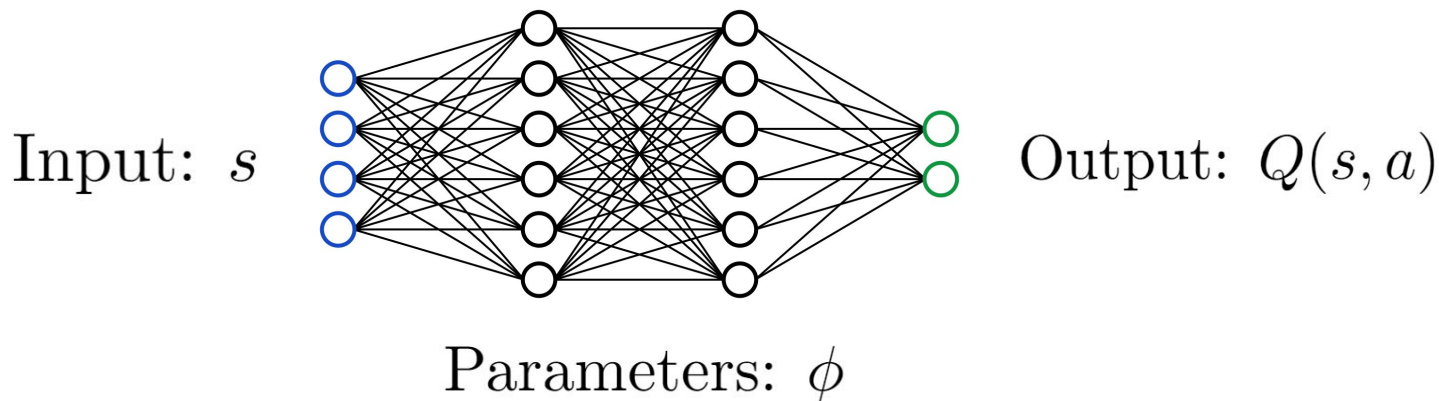
$$\text{Set } Q(s, a) \leftarrow r(s, a) + \gamma \cdot \mathbb{E}_{p(s'|s, a)} \left[V^\pi(s') \right]$$

$$\text{Set } Q(s, a) \leftarrow r(s, a) + \gamma \cdot \max_{a'} Q(s', a')$$

No longer exact. What's the approximation?

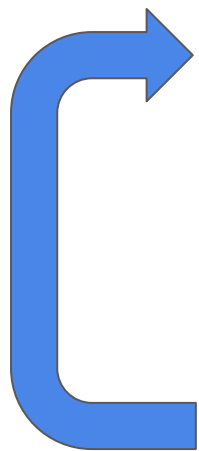
Fitted QI: Key Elements

Work with Q, instead of V



$$\arg \min_{\phi} \sum_i \frac{1}{2} \|Q_{\phi}(s_i, a_i) - y_i\|^2$$

Fitted QI: What's the Algorithm Then?



1. Collect dataset

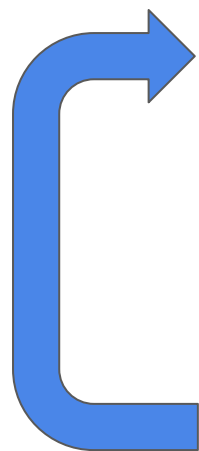


2. Set $y_i \leftarrow r(s_i, a_i) + \gamma \cdot \max_{a'_i} Q_\phi(s'_i, a'_i)$

3. Set $\phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|Q_\phi(s_i, a_i) - y_i\|^2$

What data do we need?

Fitted QI: What's the Algorithm Then?

- 
1. Collect dataset $\{(s_i, a_i, r_i, s'_i)\}$ using some policy
 2. Set $y_i \leftarrow r(s_i, a_i) + \gamma \cdot \max_{a'_i} Q_\phi(s'_i, a'_i)$
 3. Set $\phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|Q_\phi(s_i, a_i) - y_i\|^2$

How Model Free? Fitted VI vs QI

1. Collect dataset $\{(s_i, a_i, r_i, s'_i)\}$ using some policy

$$\text{Set } y_i \leftarrow \max_a \left(r(s_i, a_i) + \gamma \cdot \mathbb{E} \left[V_\phi(s'_i) \right] \right)$$

$$\text{Set } y_i \leftarrow r(s_i, a_i) + \gamma \cdot \max_{a'_i} Q_\phi(s'_i, a'_i)$$

$$\text{Set } \phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|V_\phi(s_i) - y_i\|^2$$

$$\text{Set } \phi \leftarrow \arg \min_\phi \sum_i \frac{1}{2} \|Q_\phi(s_i, a_i) - y_i\|^2$$

Let's Resurface

- Policy Iteration
- Value Iteration
- Fitted Value Iteration
- Fitted Q Iteration

MDP : Tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \rho \rangle$

\mathcal{S} : State Space

\mathcal{A} : Action Space

ρ : Initial State Distribution

$R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$: Reward Function

~~$T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$: Probabilistic Transition Function~~

Model-Free RL

Settings: Motivate Model-Free Approaches

- MDP model is unknown (no transition dyn) but we can sample from it

Autonomous vehicle in traffic

Robotic navigation in unknown envs

Advertising with unknown user behavior

- MDP model is known, but it's easier to sample

Climate models

Robotic navigation

Game playing

Monte Carlo Policy Evaluation

$$\hat{\mu} = \frac{1}{N} \sum_{i=1}^N f(x_i)$$

$$V^{\pi}(s^j) = \mathbb{E}_{p_{\theta}(\tau)} \left[\sum_{t'=t}^T \gamma^{t'-t} \cdot r(s_{t'}, a_{t'}) \middle| s_t = s^j \right]$$

Our goal: $\hat{V}^{\pi}(s^j)$

The **return** is defined as: $G_t = \sum_{t'=t}^T \gamma^{t'-t} \cdot r(s_{t'}, a_{t'})$

$$V^{\pi}(s^j) = \mathbb{E}_{p_{\theta}(\tau)} \left[G_t \middle| s_t = s^j \right]$$

How do we use Monte Carlo estimation to do policy evaluation?

Monte Carlo Policy Evaluation

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

$V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

$Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following π : $S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless S_t appears in S_0, S_1, \dots, S_{t-1} :

Append G to $Returns(S_t)$

$V(S_t) \leftarrow \text{average}(Returns(S_t))$

Targets for MC, DP and TD

$$V^\pi(s^j) = \mathbb{E}_{p_\theta(\tau)} \left[G_t \mid s_t = s^j \right]$$

Single sample return instead of real expected return

$$V^\pi(s^j) = \mathbb{E}_{p_\theta(\tau)} \left[r_{t+1} + \gamma G_{t+1} \mid s_t = s^j \right]$$

$$V^\pi(s^j) = \mathbb{E}_{p_\theta(\tau)} \left[r_{t+1} + \gamma \cdot V^\pi(s_{t+1}) \mid s_t = s^j \right]$$

True V^π not known and current estimate used instead

Incremental Approach

$$\hat{x}_m = \hat{x}_{m-1} + \alpha(m) \left(x^{(m)} - \hat{x}_{m-1} \right)$$

$$\sum_{m=1}^{\infty} \alpha(m) = \infty \quad \sum_{m=1}^{\infty} \alpha^2(m) < \infty$$

$$\hat{x} \leftarrow \hat{x} + \alpha(x - \hat{x})$$

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize} \left[\text{Target} - \text{OldEstimate} \right]$$

Incremental Model-Free Policy Evaluation

mth sample

$$\hat{V}_m^\pi(s^j) \leftarrow \hat{V}_{m-1}^\pi(s^j) + \alpha [G^{(m)} - \hat{V}_{m-1}^\pi(s^j)]$$

**Estimate at mth
iteration**

**Estimate at
m-1th iteration**

**Estimate at
m-1th iteration**

$$\hat{V}_m^\pi(s^j) \leftarrow \hat{V}_{m-1}^\pi(s^j) + \alpha [G^{(m)} - \hat{V}_{m-1}^\pi(s^j)]$$

New estimate

Old estimate

Target

Temporal Difference Policy Evaluation

$$\hat{V}_m^\pi(s^j) \leftarrow \hat{V}_{m-1}^\pi(s^j) + \alpha \left[\left(r_{t+1} + \gamma \cdot \hat{V}_{m-1}^\pi(s_{t+1}) \right)^{(m)} - \hat{V}_{m-1}^\pi(s^j) \right]$$

No expectation, single sample

No reality, bootstrap

But it works!

Algorithm: TD(0)

Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated

Algorithm parameter: step size $\alpha \in (0, 1]$

Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

$A \leftarrow$ action given by π for S

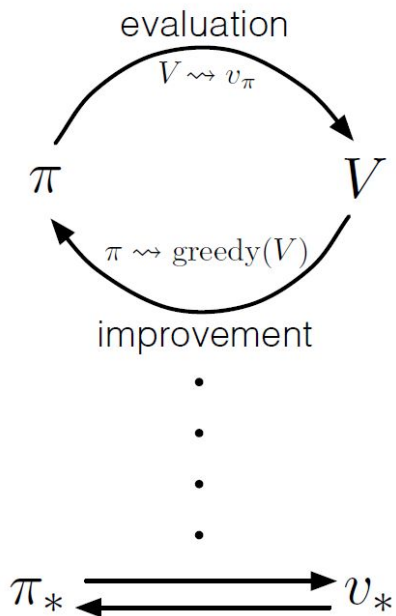
 Take action A , observe R, S'

$V(S) \leftarrow V(S) + \alpha [R + \gamma V(S') - V(S)]$

$S \leftarrow S'$

 until S is terminal

Generalised Policy Iteration



- Two simultaneous, interacting processes
 - Make value fun consistent with current policy
 - Make policy greedy w.r.t. current value function
- In PI, these processes alternate, each completing before other begins
- In VI, single iteration of policy evaluation between each policy improvement

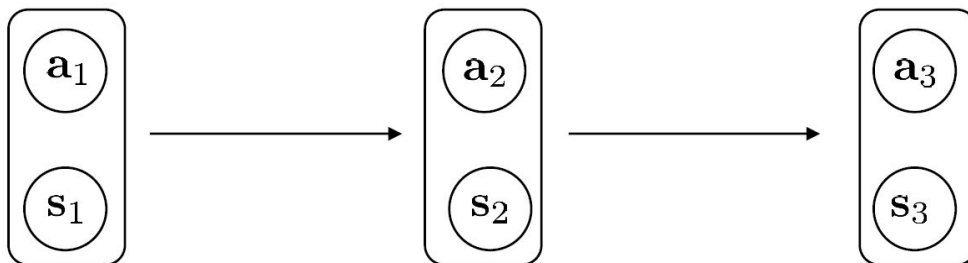
GPI: Evaluation and improvement processes interact, independent of granularity

Model-free evaluation in GPI?

Monte-Carlo Estimation of Action-Values

Why is estimating Q-values helpful towards control?

$$p((\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) | (\mathbf{s}_t, \mathbf{a}_t)) = p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t) \pi_{\theta}(\mathbf{a}_{t+1} | \mathbf{s}_{t+1})$$



ϵ – greedy policy:

For all actions to continue to be selected infinitely often, the policy has to continue to select them

$$\pi_{\text{new}} = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{A}|} & \text{if } a = \arg \max_a Q(s, a) \\ \frac{\epsilon}{|\mathcal{A}|} & \text{otherwise} \end{cases}$$

$$\pi(a|s) \geq \frac{\epsilon}{|A(s)|}$$

More general: epsilon-soft policy

Monte Carlo: Stochastic Exploration Policy

On-policy first-visit MC control (for ε -soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε -soft policy

$Q(s, a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \arg \max_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

An Issue

- Optimal policy may be deterministic
- But we need stochastic policies to visit state action pair infinitely often
- To estimate the state-action value function

Get to an almost deterministic final policy (that still explores)

On-policy type of algorithms

Use one policy to explore. Using the exploration, update another (deterministic) policy, which eventually becomes the optimal policy

Off-policy type of algorithms

Importance Sampling

$$\mathbb{E}_p [z(x)] = \int z(x)p(x)dx$$

$$r = \mathbb{E}_p [z(x)]$$

$$\hat{r} = \frac{1}{n} \sum_{i=1}^n z(x_i)$$

$$\mathbb{E}_p [z(x)] = \int z(x)p(x)dx,$$

$$= \int z(x) \frac{p(x)}{q(x)} q(x) dx$$

$$= \mathbb{E}_q \left[z(x) \frac{p(x)}{q(x)} \right]$$

$$\hat{r} = \frac{1}{n} \sum_{i=1}^n z(x_i) \frac{p(x_i)}{q(x_i)}$$

Off-Policy Prediction via Importance Sampling

$$\Pr\{A_t, S_{t+1}, A_{t+1}, \dots, S_T \mid S_t, A_{t:T-1} \sim \pi\}$$

$$= \pi(A_t | S_t) p(S_{t+1} | S_t, A_t) \pi(A_{t+1} | S_{t+1}) \cdots p(S_T | S_{T-1}, A_{T-1})$$

$$= \prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k)$$

$$\rho_{t:T-1} \doteq \frac{\prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k | S_k) p(S_{k+1} | S_k, A_k)}$$

$$= \prod_{k=t}^{T-1} \frac{\pi(A_k | S_k)}{b(A_k | S_k)}$$

Use for prediction?

Expected Returns

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{|\mathcal{T}(s)|}$$

$$V_{n+1} \doteq V_n + \frac{W_n}{C_n} [G_n - V_n], \quad n \geq 1$$

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1}}$$

$$C_{n+1} \doteq C_n + W_{n+1}$$

Can we build an incremental estimation algorithm?

Monte Carlo Prediction using Importance Sampling

Off-policy MC prediction (policy evaluation) for estimating $Q \approx q_\pi$

Input: an arbitrary target policy π

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:

$Q(s, a) \in \mathbb{R}$ (arbitrarily)

$C(s, a) \leftarrow 0$

Loop forever (for each episode):

$b \leftarrow$ any policy with coverage of π

Generate an episode following b : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

$W \leftarrow 1$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$, while $W \neq 0$:

$G \leftarrow \gamma G + R_{t+1}$

$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$

$W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$

Is this first-visit or every-visit?

Monte Carlo Control using Importance Sampling

Off-policy MC control, for estimating $\pi \approx \pi_*$

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:

$Q(s, a) \in \mathbb{R}$ (arbitrarily)

$C(s, a) \leftarrow 0$

$\pi(s) \leftarrow \operatorname{argmax}_a Q(s, a)$ (with ties broken consistently)

Loop forever (for each episode):

$b \leftarrow$ any soft policy

Generate an episode using b : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

$W \leftarrow 1$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$

$\pi(S_t) \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (with ties broken consistently)

If $A_t \neq \pi(S_t)$ then exit inner Loop (proceed to next episode)

$W \leftarrow W \frac{1}{b(A_t|S_t)}$

**Follow behavior policy
while learning about and
improving the target policy**

SARSA

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Loop for each step of episode:

 Take action A , observe R, S'

 Choose A' from S' using policy derived from Q (e.g., ε -greedy)

$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma Q(S', A') - Q(S, A)]$

$S \leftarrow S'; A \leftarrow A';$

 until S is terminal

Q-Learning

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$, arbitrarily except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

 Choose A from S using policy derived from Q (e.g., ε -greedy)

 Take action A , observe R, S'

$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$

$S \leftarrow S'$

 until S is terminal

Where Next?

Upcoming in Part 2

- Deep Q Networks
- Policy Gradients
- Actor Critic
- Exploration
- Time permitting: Why these algos work, n-step TD and Eligibility Traces, Case Studies

Announcements

- Minor Exam
 - Sunday, Sep 15, 8-10 am
 - LHC 521
- Today's office hour shifted to Friday
 - 3-4 pm
 - Room 513-D, 99-C Bldg.



[Course webpage](#)