Reinforcement Learning: Overview and Foundations

CS 6301 Special Topics: Introduction to Robot Manipulation and Navigation
Professor Yu Xiang
The University of Texas at Dallas
Reinforcement Learning

Reinforcement Learning:

Imitation Learning:

\[ a_t = \pi(S_t) \]
RL Examples

Control

RL Concepts

• State $s$: a complete description of the state of the world

• Observation $o$: partial description of a state
  • Fully observed vs. partially observed

• Action space: the set of all valid actions in a given environment
  • Discrete action space vs. continuous action space

• Policies: a policy is a rule used by an agent to decide what action to take
  • Deterministic policy $a_t = \mu(s_t)$
  • Stochastic policy $a_t \sim \pi(\cdot|s_t)$
RL Concepts

• Parameterized policies
  \[ a_t = \mu_\theta(s_t) \]
  \[ a_t \sim \pi_\theta(\cdot | s_t) \]

• Deterministic policy

• Stochastic policy
  • Categorical policy for discrete actions
    \[ \log \pi_\theta(a | s) = \log [P_\theta(s)]_a \]
  • Diagonal Gaussian policy: mean action
    \[ \mu_\theta(s) \]
  Log standard deviation
    \[ \log \sigma_\theta(s) \]
RL Concepts

• Diagonal Gaussian policy
  • Sampling \( a = \mu_\theta(s) + \sigma_\theta(s) \odot z \) \( z \sim \mathcal{N}(0, I) \)

• Log-likelihood

\[
\log \pi_\theta(a|s) = -\frac{1}{2} \left( \sum_{i=1}^{k} \left( \frac{(a_i - \mu_i)^2}{\sigma_i^2} + 2 \log \sigma_i \right) + k \log 2\pi \right)
\]
RL Concepts

• A Trajectory is a sequence of states and actions in the world
  \[ \tau = (s_0, a_0, s_1, a_1, \ldots) \]

• Start-state distribution
  \[ s_0 \sim \rho_0(\cdot) \]

• State transitions are governed by natural laws of the environment
  • Deterministic
    \[ s_{t+1} = f(s_t, a_t) \]
  • Stochastic
    \[ s_{t+1} \sim P(\cdot|s_t, a_t) \]
RL Concepts

• Reward function

\[ r_t = R(s_t, a_t, s_{t+1}) \]  
\[ r_t = R(s_t) \]  
\[ r_t = R(s_t, a_t) \]

• Finite-horizon undiscounted return

\[ R(\tau) = \sum_{t=0}^{T} r_t \]

• Infinite-horizon discounted return

\[ R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t \]  
\[ \gamma \in (0, 1) \]
The RL Problem

• The goal of RL is to select a policy which maximizes expected return when the agent acts according to it

• Probability distribution over trajectories

\[ P(\tau|\pi) = \rho_0(s_0) \prod_{t=0}^{T-1} P(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \]

• Expected return

\[ J(\pi) = \int P(\tau|\pi)R(\tau) = \mathbb{E}_{\tau \sim \pi} [R(\tau)] \]

• The central optimization problem

\[ \pi^* = \arg \max_{\pi} J(\pi) \]

Optimal policy
Value Functions

• Value of a state or a state-action pair
  • The expected return if you start in that state or state-action pair, and then act according to a particular policy forever after

• On-policy Value Function
  \[ V^\pi(s) = \mathbb{E}_{\tau \sim \pi} [R(\tau) \mid s_0 = s] \]

• On-policy Action-Value Function
  \[ Q^\pi(s, a) = \mathbb{E}_{\tau \sim \pi} [R(\tau) \mid s_0 = s, a_0 = a] \]

• Optimal Value Function
  \[ V^*(s) = \max_{\pi} \mathbb{E}_{\tau \sim \pi} [R(\tau) \mid s_0 = s] \]

• Optimal Action-Value Function
  \[ Q^*(s, a) = \max_{\pi} \mathbb{E}_{\tau \sim \pi} [R(\tau) \mid s_0 = s, a_0 = a] \]
Value Functions

• Connection

\[ V^\pi(s) = \mathbb{E}_{a \sim \pi} \left[ Q^\pi(s, a) \right] \]

\[ V^*(s) = \max_a Q^*(s, a) \]

• The optimal policy in \( s \) will select whichever action maximizes the expected return starting in \( s \)

\[ a^*(s) = \arg \max_a Q^*(s, a) \]
Bellman Equations

- The value of your starting point is the reward you expect to get from being there, plus the value of wherever you land next

\[ V^\pi(s) = \mathbb{E}_{a \sim \pi, s' \sim P} \left[ r(s, a) + \gamma V^\pi(s') \right], \]

On-policy

\[ Q^\pi(s, a) = \mathbb{E}_{s' \sim P} \left[ r(s, a) + \gamma \mathbb{E}_{a' \sim \pi} [Q^\pi(s', a')] \right] \]

Optimal policy

\[ V^*(s) = \max_a \mathbb{E}_{s' \sim P} \left[ r(s, a) + \gamma V^*(s') \right], \]

\[ Q^*(s, a) = \mathbb{E}_{s' \sim P} \left[ r(s, a) + \gamma \max_{a'} Q^*(s', a') \right] \]
Advantage Functions

• How much better it is to take a specific action $a$ in state $s$, over randomly selecting an action according to $\pi(\cdot | s)$

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$
Markov Decision Processes (MDPs)

• A MDP is a 5-tuple $\langle S, A, R, P, \rho_0 \rangle$

• $S$ is the set of all valid states,

• $A$ is the set of all valid actions,

• $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function, with $r_t = R(s_t, a_t, s_{t+1})$,

• $P : S \times A \rightarrow \mathcal{P}(S)$ is the transition probability function, with $P(s'|s, a)$ being the probability of transitioning into state $s'$ if you start in state $s$ and take action $a$,

• and $\rho_0$ is the starting state distribution.
A Taxonomy of RL Algorithms

RL Algorithms

Model-Free RL
- Policy Optimization
  - Policy Gradient
    - A2C / A3C
    - PPO
    - TRPO
  - DDPG
  - TD3
  - SAC

Model-Based RL
- Q-Learning
  - DQN
  - C51
  - QR-DQN
  - HER

- Learn the Model
  - World Models
    - I2A
    - MBMF
  - Given the Model
    - AlphaZero
Model-Free vs. Model-based RL

• Whether the agent has access to (or learns) a model of the environment

• A model is a function which predicts state transitions and reward

• A model allows the agent to plan by thinking ahead

• A ground-truth model of the environment is usually not available to the agent
Model-Free RL

- **Policy optimization**
  - Represent a policy as $\pi_\theta(a|s)$
  - Optimize the parameters $\theta$ by gradient descent
  - Optimization is **on-policy**: update only uses data collected while acting according to the most recent version of the policy

- **Q-Learning**
  - Learns an approximator $Q_\theta(s, a)$ for the optimal action-value function $Q^*(s, a)$
  - Optimization is **off-policy**: each update can use data collected at any point during training (sample efficient)

$$a(s) = \arg \max_a Q_\theta(s, a)$$
Model-based RL

• How to use the model?
• Pure planning: model-predictive control (MPC)
• Expert iteration
  • uses a planning algorithm (like Monte Carlo Tree Search) in the model
  • The policy is updated to produce an action more like the planning algorithm’s output
  • [https://www.deepmind.com/blog/alphago-zero-starting-from-scratch](https://www.deepmind.com/blog/alphago-zero-starting-from-scratch)
• Data augmentation for model-free methods
• Embedding planning loop into policies
  • The policy can learn to choose how and when to use the plans
Summary

• RL concepts

• Model-free vs. model-based methods
Further Reading

• OpenAI Spinning Up in Deep RL