DRL Homework 01

Group 27

April 22, 2022

Contents

1 Task 1

Game of chess as an MDP

$$MDP(S, A, p(S_{t+1}|S_t, A_t), r(S_t, A_t))$$

This is a set of all chess pieces:

$$P := \{p \mid \text{p is one of the possible chess pieces}\}$$

This is the set of all states:

$$S := \{x \in (P \cup \{empty\})^{8\times 8} | \text{x is one of the possible chess combination of pieces and empty fields}\}$$

This is the initial state:
$S_0 \doteq (rook, knight, bishop, queen, king, bishop, knight, rook,$
$pawn, pawn, pawn, pawn, pawn, pawn, pawn, pawn, empty, ...,$
$empty, pawn, pawn, pawn, pawn, pawn, pawn, pawn, pawn,$
$rook, knight, bishop, king, queen, bishop, knight, rook)$
This is the set of possible actions:

$$A \doteq \{a \in (i, j) \mid 0 \leq i, j \leq 63, \text{ where a piece from i is moved to j}\}$$

This is the reward function:

$$r(S_t, A_t) \doteq \sum_{i=0} pieceVal(S_{(t+1)_i}) - \sum_{i=0} pieceVal(S_{t_i})$$

$$pieceVal(p) \doteq \begin{cases} 100 & \text{if own king} \\ -100 & \text{if opposing king} \end{cases}$$

This is the environment dynamics:

$$p : S \times S \times A \to \mathbb{R}; (S_{t+1}, S_t, A_t) \mapsto \begin{cases} 1 & \text{if } S_{t+1} \text{ is optimal as det. by a Minimax implementation} \\ 0 & \text{else} \end{cases}$$

This is the Policy:

$$\pi : A \times S \to \mathbb{R}; (A_t, S_t) \mapsto \frac{\sum_{S_{t+1}}[p(S_{t+1}|S_t, A_t)(r(S_{t+1}, A_t) + \gamma V_\pi(S_{t+1}))]}{\sum_{a \in A} \sum_{S_{t+1}}[p(S_{t+1}|S_t, a)(r(S_{t+1}, a) + \gamma V_\pi(S_{t+1}))]}$$

2 Task 2

Lunar Lander as an MDP
$MDP(S, A, p(S_{t+1}|S_t, A_t), r(S_t, A_t))$
$C \doteq \{(x, y) :$ (x,y) are the coordinates of lander$\}$
$V \doteq \{(\vec{x}, \vec{y}) : (\vec{x}, \vec{y})$ are linear velocities in x and y of lander$\}$
$A \doteq \{\alpha : \alpha$ is angle of lander$\}$
$V_A \doteq \{\vec{\beta} : \vec{\beta}$ is angular velocity of lander$\}$
$G \doteq \{(ll, lr) :$ (ll,lr) booleans for each leg if it touches the ground$\}$
These are the possible states
$S \doteq \{s \in C \times V \times A \times V_A \times G :$ s is one possible state of the lander$\}$
These are the possible actions
$A \doteq \{a \in \{no, left, main, right\} :$ a is one of 4 possible actions(no action, fire left, fire middle, fire
$p(S_{t+1}|S_t, A_t)$ is determined by physics
$r(s, a) \doteq legs(S_{t+1}) + ground(S_{t+1}) + location(S_{t+1}) + engines(a)$

$$legs(s) \doteq \begin{cases} 10 & \text{if one leg touches the ground} \\ 20 & \text{if two legs touch the ground} \end{cases}$$

$$ground(s) \doteq \begin{cases} 100 & \text{if } legs(s) = 20 \wedge \vec{\beta} = 0 \wedge (\vec{x}, \vec{y}) = (0, 0) \\ -100 & \text{if } legs(s) = 20 \wedge \vec{\beta} \neq 0 \wedge (\vec{x}, \vec{y}) \neq (0, 0) \end{cases}$$

$location(s) \doteq -x - y$

$$engines(a) \doteq \begin{cases} -0.3 & \text{if } a = main \\ -0.03 & \text{if } a = left \vee a = right \end{cases}$$

This is the Policy:

$$\pi : A \times S \to \mathbb{R}; (A_t, S_t) \mapsto \frac{\sum_{S_{t+1}} [p(S_{t+1}|S_t, A_t)(r(S_{t+1}, A_t) + \gamma V_\pi(S_{t+1}))]}{\sum_{a \in A} \sum_{S_{t+1}} [p(S_{t+1}|S_t, a)(r(S_{t+1}, a) + \gamma V_\pi(S_{t+1}))]}$$

$optional.Rewards =$ Reward for moving from the top of the screen to the landing pad and coming to rest is about 100-140 points.
If the lander moves away from the landing pad, it loses reward.
If the lander crashes, it receives an additional -100 points.
If it comes to rest, it receives an additional +100 points.
Each leg with ground contact is +10 points.
Firing the main engine is -0.3 points each frame.
Firing the side engine is -0.03 points each frame. Solved is 200 points.

3 Task 3

The environment dynamics is defined as

$$p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$$

for all s', $s \in \mathcal{S}$, $r \in \mathcal{R}$, and $a \in \mathcal{A}$ (Sutton and Barto 2018, p. 48). The Markov property defines that the state and reward values depend only on the immediately preceding state and action rather than the entire history. The information required to determine what the next state and reward will be when the agent takes an action should be contained within the current state.

In real-world problems, it is rare that we have complete knowledge of the environment dynamics, i.e. an all-encompassing model of the environment is normally not readily available, but rather, we may have a limited dataset of observations made in the environment, or we may collect observations in order to obtain such a dataset.

In a real-world scenario the dynamics may also not be completely known, e.g. in the scenario of a self-driving car, not all behavior of real-world drivers can be predicted with a mathematical model.

References

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). The MIT Press. http://incompleteideas.net/book/the-book-2nd.html