強化学習まとめ

用語整理

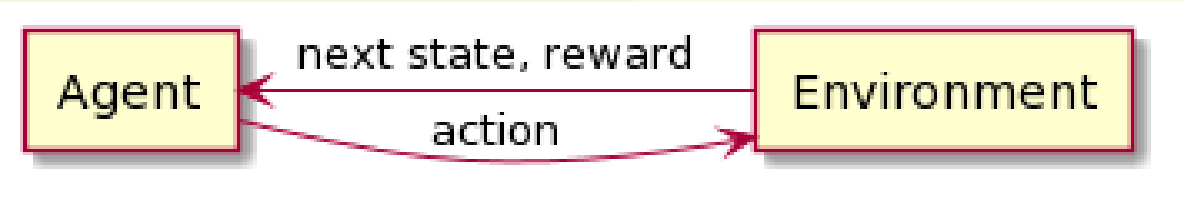
- モデルベース：環境モデルの情報がわかっている。また、未知であっても、観測データから環境モデルを学習して活用する手法も含む。
  - 動的計画法：ベルマン方程式を解析的に解く
    - 方策反復法
    - 価値反復法

用語整理

- モデルフリー：環境モデルの知識を前提としない学習法
  - 価値ベース
    - Q学習
    - SARSA
  - 方策ベース
    - 方策勾配法
    - Actor-Critic（ハイブリッド）

用語整理

- 状態 s_t：時刻 t におけるシステムの状態
- 行動 a_t：時刻 t にエージェントの選択する行動
- 報酬 R_{t+1}：行動 a_t によってエージェントに与えられる報酬
- 方策 $\pi(a|s)$：状態 s で行動 a を選択する確率
- 遷移確率 $p(s'|s,a)$：状態 s で行動 a を選択したとき、状態 s' に遷移する確率
- 割引報酬和（収益、return）：$\sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k} (0 < \gamma < 1)$



例

- 状態：$Home, Office, Bar$（終端状態なし）
- 行動：$move, stay$
- 報酬：例えば、 $r(Home, move, Bar) = +2$
- 方策：例えば、 $\pi(stay|Office) = 0.5$
- 遷移確率：例えば、
  $p(Office|Home, move) = 0.8$
  $p(Bar|Home, move) = 0.2$

価値の定量化

- 状態価値関数 $v_\pi(s)$
  - 状態 s における価値
  - 「状態 s を起点に、方策 π に従って行動したときの報酬の期待値」と定義する
- 行動価値関数 $q_\pi(s, a)$
  - 状態 s で行動 a を採る価値
  - 「状態 s で行動 a を選択した後、方策 π に従って行動したときの報酬の期待値」と定義する

（状態）価値関数の定式化

- 1ステップから期待される収益は以下で表される

$$r(s, a, s') + \gamma v_\pi(s')$$

- 方策と遷移確率を考慮して期待値を計算すると、価値関数が導かれる（Bellman方程式）

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|s, a)\{r(s, a, s') + \gamma v_\pi(s')\} \cdots (1)$$

（行動）価値関数の定式化

- (1)から $\pi(a|s)$ をとる

$$q_\pi(s,a) = \sum_{s'} p(s'|s,a)\{r(s,a,s') + \gamma v_\pi(s')\} \cdots (2)$$

- (1)(2)より、$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a) \cdots (3)$
- (2)(3)より、

$$q_\pi(s,a) = \sum_{s'} p(s'|s,a)\{r(s,a,s') + \gamma \sum_{a'} \pi(a'|s') q_\pi(s',a')\}$$

最適ベルマン方程式

- 状態 s において最適な方策をとったときの価値関数

$$v_*(s) = \max_\pi v_\pi(s)$$

$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

と定義すると、以下で表される。**最適な行動は複数のときもある。**

$$v_*(s) = \max_{a \in A^*(s)} \sum_{s'} p(s'|s,a)\{r(s,a,s') + \gamma v_*(s')\}$$

動的計画法

- 方策反復法
  - 方策評価と方策改善の繰り返しで最適方策を得る。
- 価値反復法
  - 方策反復法ではベルマン方程式を解く計算コストが高いので、代わりに最適ベルマン方程式を解く。

方策反復法（Policy Iteration）

方策評価ステップ

- (1) より以下のベクトル方程式が導かれる。

$$\boldsymbol{v} = \boldsymbol{R}^{\boldsymbol{\pi}} + \gamma \boldsymbol{P}^{\boldsymbol{\pi}} \boldsymbol{v} \rightarrow \boldsymbol{v} = (1 - \gamma \boldsymbol{P}^{\boldsymbol{\pi}})^{-1} \boldsymbol{R}^{\boldsymbol{\pi}}$$

ただし、
$[P^{\pi}]_{ss'} = \sum_a \pi(a|s) p(s'|s,a)$
$[R^{\pi}]_s = \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) r(s,a,s')$

方策反復法（Policy Iteration）

方策改善ステップ

$$\pi'(a|s) = \begin{cases} 1/|A^*(s)| & (for a \in A^*(s)) \\ 0 & (otherwise) \end{cases}$$

ただし、$A^*(s) = a_* s.t. a_* = \arg\max_a q_\pi(s,a)$

- $q_\pi(s,a)$ は(2)で計算する。

価値反復法（Value Iteration）

- 価値関数の更新

$$v_{t+1}(s) = \max_a q_{t+1}(s, a)$$

$$v_0(s) = 0 (\text{この初期値はあくまで例})$$

- 方策の更新

$$\pi_{t+1}(s) = \arg\max_a q_{t+1}(s, a)$$

ただし、$q_{t+1}(s, a) = \sum_{s'} p(s'|s, a)\{r(s, a, s') + \gamma v_t(s')\}$

モンテカルロ法

- 終端に至るまでの状態報酬系列 $\{S_t, R_t | t = 0, 1, \cdots, T\}$ をサンプリングして、そのときの収益 G の平均値を推定値 V として求める。
- エピソードが完結するまで計算できない。
- バイアス小、バリアンス大。

モンテカルロ法

- **サンプリング時に S_t, R_t （とそこから導かれる G_t）はわかっている**ため、以下を計算できる。

$$V_{t+1}(s) = \frac{1}{N_{t+1}(s)} \sum_{k=0}^{t} G_k \mathbf{1}(S_k = s) \cdots (1)$$
$$N_{t+1}(s) = \sum_{k=0}^{t} \mathbf{1}(S_k = s) \cdots (2)$$

(1)(2)より、以下が導かれる（ただし、$V_0(s) \equiv 0, N_0(s) \equiv 0$）

$$V_{t+1}(s) = V_t(s) + \frac{1}{N_{t+1}(s)} \{G_t - V_t(S_t)\} \mathbf{1}(S_t = s)$$
$$N_{t+1} = N_t(s) + \mathbf{1}(S_t = s)$$

モンテカルロ法

ここで、$\frac{1}{N_{t+1}(s)} = \alpha_t$（学習率）と一般化すると、上式は推定値$V_t(S_t)$を目標値$G_t$に近づけていると解釈できる。

$$V_{t+1}(s) = V_t(s) + \alpha_t\{G_t - V_t(S_t)\}\mathbf{1}(S_t = s)$$

- α_tがRobbins-Monro条件を満たせば、$V_t(s)$の収束性は保証されるが、実用上は十分に小さい$\alpha_t = \alpha$（定数）として問題ない。

TD 学習

メモ

- 行動を表す変数が高次元だったり連続だったりすると、Q学習が難しくなる→行動価値を推定するよりも行動の確率分布を記述する方策を学習するほうが有効→方策勾配法、Actor-Critic
-