強化学習の基礎

小池　康晴 †, 鮫島 和行 ‡

† 東京工業大学 精密工学研究所
‡ 科学技術振興事業団 ERATO 川人学習動態脳プロジェクト

1　教師無し学習

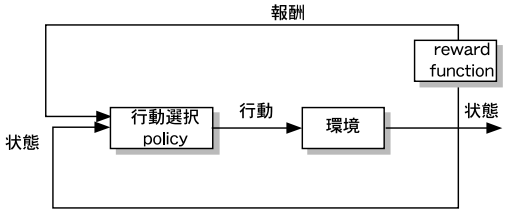
赤ちゃんが生れてから、手を動かしたり、物をつかんだりするとき、手の動かし方や、物のつかみ方を直接教えてもらうことなく、環境との相互作用により、自然に学習する。このような学習は、教師なし学習とよばれている。一方、絵や文字を覚えるような学習は、覚える絵や文字が与えられているため、教師あり学習と呼ばれる。しかし、これらは、完全に分離することはできない。例えば、言葉を覚えるとき、自分の発生した音と他人が発生した音は、直接比較することができるため、その差をつかって教師あり学習をすることができる。しかし、調音器官の筋肉一本一本の動きは直接知ることができないため、教師無し学習を行うことになる。

近年、目標パターン（この例では言葉）は与えられずに、出力が良かったか、悪かったかだけ（この例では、上手く発音できたらほめられる）を与えるだけで、その評価を最大にする出力を学習する枠組みとして強化学習（Reinforcement Learning）が提案されている。

2　強化学習の要素

まずはじめに、強化学習の枠組みを間単に説明する。強化学習では、様々な行動を試してみて、より良い報酬が得られる行動を選択することを行う。

- policy
  行動を決めるルール
- reward function
  ある状態において獲得できる報酬
- value function
  将来にわたって獲得できる報酬の総和



強化学習の他の学習則と異る特徴は、正しい行動を教えられるのではなく、取った行動を評価することで学習をおこなうことである。そのために、二つのプロセスを繰り返しおこなう。一つは、現在のポリシーに従って行動をおこない、得られた報酬から価値関数 (Value function) を作成すること（ポリシーの評価）であり、もう一つは、現在の価値関数から最適なポリシーを作成すること（ポリシーの改善）である。

3　環境との相互作用による学習

強化学習では、環境から得られる最終的な累積報酬を最大化することで学習を行う。累積報酬は、以下の式で与えられる。

$$R_t = \sum_{k=0}^{T} \gamma^k r_{t+k+1}$$

ここで、T は、最終時刻、γ は、遠い将来に得られる報酬ほど割引いて評価するための割引き率 (discount factor) であり、$0 \leq \gamma \leq 1$ である。

3.1　マルコフ性

ここで、マルコフ性の定義をおこなう。数学的記述の簡単化のために状態と、報酬の数は有限であると仮定する。時刻 t で取った行動に対して、時刻 $t+1$ において、どのように応答するかを考える。一般的には、時刻 $t+1$ 以前に起った全ての事象が関係するため、次のように定義される。

$$Pr\{s_{t+1} = s', r_{t+1} = r | s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0\}$$

一方、状態信号がマルコフ性を持つならば、$t+1$ の応答は一時刻前 t の状態と行動だけによって決る。

$$Pr\{s_{t+1} = s', r_{t+1} = r | s_t, a_t\}$$

このように、マルコフ性を仮定すると、現在の状態と行動から次の時刻の状態と報酬を予測することができる。さらに、繰り返し計算により、すべての将来の状態と報酬を予測することができる。

マルコフ性を満足する強化学習は、マルコフ決定過程 (Markov decision process:MDP) と呼ばれる。有限 MDP では、任意の状態 s と行動 a が与えられると、可能な次の状態 s' の確率は、次の式で与えられる。

$$\mathcal{P}^a_{ss'} = Pr\{s_{t+1} = s' | s_t = s, a_t = a\}$$

これは、遷移確率 (transition probabilities) と呼ばれる。また、現在の状態 s と行動 a が与えられたとき次の状態 s' での報酬の期待値は、以下のようになる。

$$\mathcal{R}^a_{ss'} = E\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\}$$

3.2　価値関数 (*value function*)

強化学習では、報酬を評価してその評価を最大化することで学習を行う。ここでは、現在の状態（あるいは、行動）がどのくらい良いのか、を計る関数として、価値関数というものを考える。“どのくらい良いのか” ということを、将来にわたって得られる報酬によって定義する。

方策 π というのは、状態 $s \in \mathcal{S}$ で行動 $a \in \mathcal{A}(s)$ をとることであり、$\pi(s, a)$ と表す。方策 π のもとで、状態 s の価値は、以下のように定式化できる。

状態価値関数 *(state-value function for policy π)*

$$V^\pi(s) = E_\pi\{R_t | s_t = s\} = E_\pi\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\}$$

同様に、方策 π のもとで、状態 s において行動 a を取ることの価値は、以下のように定義できる。

行動価値関数 *(action-value function for policy π)*

$$Q^\pi(s,a) = E_\pi\{R_t|s_t = s, a_t = a\} = E_\pi\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s, a_t = a\}$$

価値関数は、以下のような再帰的な関係をもっている。*Bellman equation*

$$\begin{aligned}
V^\pi(s) &= E_\pi\{R_t|s_t = s\} \\
&= E_\pi\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s\} \\
&= E_\pi\{r_{t+1} + \sum_{k=0}^{\infty}\gamma^k r_{t+k+2}|s_t = s\} \\
&= \sum_a \pi(s,a)\sum_{s'}\mathcal{P}^a_{ss'}\{\mathcal{R}^a_{ss'} + \gamma E_\pi\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+2}|s_{t+1} = s'\} \\
&= \sum_a \pi(s,a)\sum_{s'}\mathcal{P}^a_{ss'}\{\mathcal{R}^a_{ss'} + \gamma V^\pi(s')\}
\end{aligned}$$

この式は、すべての行動に対して、すべての期待される次の状態での（ディスカウントされた）価値と報酬の和を生起確率で重みづけしたものである。

3.3 Optimal Value Functions

評価することが出来るようになったため、次に行うことは、最適な価値関数を求めることである。これは、もっとも報酬を多く得られる方策を求めることである。

最適な状態価値関数を V^* で表すと、

$$V^*(s) = \max_\pi V^\pi(s)$$

となる。また、最適な行動価値関数を Q^* で表すと、

$$Q^*(s,a) = \max_\pi Q^\pi(s,a)$$

となる。これは、状態と行動のペア (s,a) に対して、状態 s において、行動 a をとり、以後、最適な方策に従って行動したときに得られる報酬の期待値である。したがって、Q^* は、V^* を用いて以下の様に書くことができる。

$$Q^*(s,a) = E_\pi\{r_{t+1} + \gamma V^*(s_{t+1})|s_t = s, a_t = a\}$$

V^* に対する最適な Bellman 方程式は、以下のようにかける。

$$\begin{aligned}
V^*(s) &= \max_{aA(s)} Q^{\pi^*}(s,a) \\
&= \max_a E_{\pi^*}\{R_t|s_t = s, a_t = a\} \\
&= \max_a E_\pi\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s, a_t = a\} \\
&= \max_a E_{\pi^*}\{r_{t+1} + \sum_{k=0}^{\infty}\gamma^k r_{t+k+2}|s_t = s, a_t = a\} \\
&= \max_a E\{r_{t+1} + \gamma V^*(s_{t+1})|s_t = s, a_t = a\} \\
&= \max_a \sum_{s'}\mathcal{P}^a_{ss'}\{\mathcal{R}^a_{ss'} + \gamma V^*(s')\}
\end{aligned}$$

また、Q^* に対する最適な Bellman 方程式は、以下のようである。

$$\begin{aligned} Q^*(s,a) &= E\{r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') | s_t = s, a_t = a\} \\ &= \sum_{s'} P^a_{ss'} \left(R^a_{ss'} + \gamma \max_{a'} Q^*(s', a') \right) \end{aligned}$$

4 Temporal-Difference Learning

TD 学習は、Monte Carlo 法と Dynamic Programming の組合せと考えることができる。環境のモデルを使わず経験的に学習をおこなう、という点で MC 法に、また、最終結果を待たずに、評価を途中で更新する点で DP 法に似ている。

Monte Carlo 法では、各時刻の報酬が分るまで待ち価値関数を更新するため、単純な Monte Carlo 法では、価値関数の更新は次のようになる。

$$V(s_t) \leftarrow V(s_t) + \alpha[R_t - V(s_t)]$$

一方 TD 法では、次のステップを待つだけで、価値関数を更新する。

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

$r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$ は、TD 誤差と呼ばれている。

4.1 Q-learning

off-policy TD → Q-learning

各状態において、可能な行動の中で最も行動評価関数の値が高い行動をとるように学習を行う方法を Q Learning と呼ぶ。その学習は、

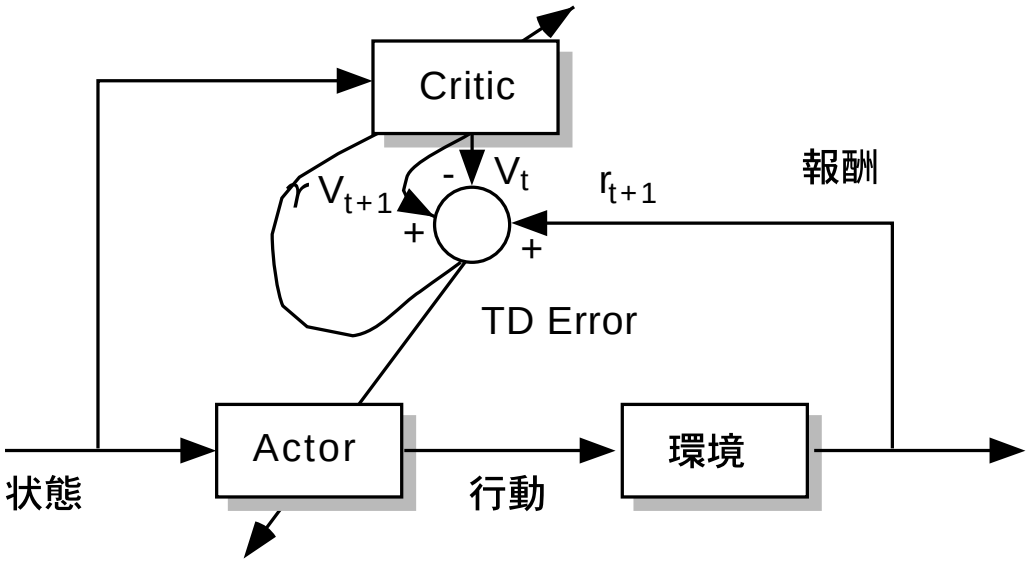
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

というように行われる。Q Learning が方策 off 型と呼ばれるのは、方策に関係なく行動価値関数の最大値で行動価値関数を更新するためである。

4.2 Actor-Critic

on-policy TD → Actor-Critic

Actor-Critic 法は、価値関数とは独立に、方策を表現する構造を別に持っている。これは、行動を選択するために用いられるため “actor” と呼ばれている。また、価値関数を予測する部分は、actor によって選ばれた行動を批判するために “critic” と呼ばれている。



状態価値関数は、

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

によって更新される。

TD 誤差は、行った行動を評価する為に用いられる。TD 誤差が正の値ならば、行った行動は価値を高めたことになるので、より選択されるように、逆に、TD 誤差が負の値ならば、選択されないようにするほうがよい。

$p(s,a)$ を actor が状態 s で行動 a を取る確率だとすると、行動選択は、

$$\pi_t(s,a) = Pr\{a_t = a | s_t = s\} = \frac{e^{p(s,a)}}{\sum_b e^{p(s,b)}}$$

で表される。

先に示したように actor は、

$$p(s_t, a_t) \leftarrow p(s_t, a_t) + \beta\delta_t$$

として、学習をおこなう。ここで、β は正のステップサイズ変数であり、δ_t は TD 誤差である。

4.3 TD(λ)

これまでみてきた TD 法は、1 ステップ後での評価を比較して価値関数を更新していた。ここでは、n ステップ後まで考えるように拡張をおこなう。

n ステップ後までの報酬を考慮にいれると、最終的に得られる累積報酬は以下のようになる。

$$R_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V_t(S_{t+n})$$

さらに、任意の n ステップの報酬だけでなく、n ステップの報酬の平均値を考えることもできる。

TD(λ) は、n ステップのバックアップを平均化する方法の一つである。

$$R_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)}$$

ここでは、$\lambda^{n-1}, 0 \leq \lambda \leq 1$ によって重み付けされている。$(1-\lambda)$ は、正規化のための項である。

実際にアルゴリズムを実装するためには、逆の見方をするほうが分りやすい。そこで、*eligibility trace* という考え方を導入する。ある時刻 t で状態 s のとき、*eligibility trace* $e_t(s)$ は、以下のように定義できる。

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & if s \neq s_t; \\ \gamma\lambda e_{t-1}(s) + 1 & if s = s_t; \end{cases}$$

これを用いると、価値関数の更新式は以下のようになる。

$$\Delta V_t(s) = \alpha\delta_t e_t(s), \text{for all } s \in \mathcal{S}$$

5 脳のモデル

近年、サルの条件反射学習の実験において、大脳基底核のドーパミンニューロンは、学習前は、報酬そのものに反応するが、学習が進むにつれて、報酬を予測させる刺激に対して反応するようになることが示された。

この反応は、TD 誤差の振舞いと良く似ているため、強化学習で用いられる、状態価値関数や、行動価値関数のような出力を持つ細胞が脳のなかに獲得されている可能性もある。