Markov decision process

Definition

A Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ consists of

- $\mathcal{S}$ a set of states (in my case $\subseteq \mathbb{R}^n$)
- $\mathcal{A}$ a set of actions (in my case *finite*)
- $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ its Markov transition kernel
- $R : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathbb{R})$ its immediate reward distribution
- $\gamma \in (0, 1)$ the discount factor

Q-Learning

- Policy: $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$
- State-value function: $V^\pi : \mathcal{S} \to \mathbb{R}$
  $V^\pi(s) = \mathbb{E}\left(\sum_{t \geq 0} \gamma^t R_t \mid R_t \sim R(S_t, A_t), S_t \sim P(S_{t-1}, A_{t-1}), A_t \sim \pi(S_t), S_0 = s\right)$
- State-action-value (Q-) function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

$$Q^\pi(s, a) = \mathbb{E}(R(s, a) + \gamma V^\pi(S_0) \mid S_0 \sim P(s, a))$$

- Optimal Q-function is defined as

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

- One can show that there is a policy π^* such that $Q^* = Q^{\pi^*}$. This is the optimal policy - the goal of Q-learning (and Reinforcement Learning in general).

Artificial Neural Networks

Definition

An **ANN** (Artificial Neural Network) with structure $\{d_i\}_{i=0}^{L+1} \subseteq \mathbb{N}$, activation functions $\sigma_i = (\sigma_{ij} : \mathbb{R} \to \mathbb{R})_{j=1}^{d_i}$ and weights $\{W_i \in M^{d_i \times d_{i-1}}, v_i \in \mathbb{R}^{d_i}\}_{i=1}^{L+1}$ is the function $F : \mathbb{R}^{d_0} \to \mathbb{R}^{d_{L+1}}$

$$F(x) = w_{L+1} \circ \sigma_L \circ w_L \circ \sigma_{L-1} \circ \cdots \circ w_1 x$$

where w_i is the affine function $x \mapsto W_i x + v_i$ for $i \in [L+1]$.
Here $\sigma_i(x_1, \ldots, x_{d_i}) = (\sigma_{i1}(x_1), \ldots, \sigma_{id_i}(x_{d_i}))$.
$L \in \mathbb{N}_0$ is called the number of hidden layers.
d_i is the number of neurons or nodes in layer i.

An ANN is called *deep* if there are two or more hidden layers.

The Bellman operator

Denote π_Q as the *greedy* policy with respect to Q i.e. $\pi(s, a) = 1$ for $a = \text{argmax}_a Q(s, a)$. For every policy π we define the operators

$$(P^\pi Q)(s, a) = \mathbb{E}(Q(S', A') \mid S' \sim P(s, a), A' \sim \pi(S'))$$

$$(T^\pi Q)(s, a) = \mathbb{E}R(s, a) + \gamma(P^\pi Q)(s, a)$$

T^π is called the Bellman operator. It can be shown that Q^π is a fixed point for T^π. Finally we define *Bellmans optimality operator* T as

$$TQ = T^{\pi_Q} Q$$

Bellmans optimality equation is then $TQ^* = Q^*$.

Context

Theorem

If both $\mathcal{S}$ and $\mathcal{A}$ are finite, and R is deterministic, then the simple iteration

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t(s_t, a_t)[R_t + \gamma Q_t(s_{t+1}, \pi_{Q_t}(s_{t+1})) - Q_t(s_t, a_t)]$$

converges with probability 1 to Q^*, given that

$$\sum_{t \geq 1} \alpha_t(s, a) = \infty, \qquad \sum_{t \geq 1} \alpha_t^2(s, a) < \infty$$

for all $(s, a) \in \mathcal{S} \times \mathcal{A}$.

Theorem (Universal approximation theorem)

An ANN with 1 hidden layer is sufficient to approximate any continuous function $[0, 1]^k \to \mathbb{R}$ (at cost of layer-width).

Sparse ReLU Network

Definition

For $s, V \in \mathbb{R}$ a (s,V)-**Sparse ReLU Network** is an ANN f with any structure $\{d_i\}_{i\in[L+1]}$, all activation functions being $ReLU$ i.e. $\sigma_{ij} = \max(\cdot, 0)$ and any weights (W_ℓ, v_ℓ) satisfying

- $\max_{\ell\in[L+1]} \left\| \widetilde{W_\ell} \right\|_\infty \leq 1$
- $\sum_{\ell=1}^{L+1} \left\| \widetilde{W_\ell} \right\|_0 \leq s$
- $\max_{j\in[d_{L+1}]} \|f_j\|_\infty \leq V$

Here $\widetilde{W_\ell} = (W_\ell, v_\ell)$.

The set of them we denote $\mathcal{F}(s, V)$.

The algorithm

Algorithm 1: Fitted Q-Iteration Algorithm

Input: MDP $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, function class $\mathcal{F}$, sampling distribution ν, number of iterations K, number of samples n, initial estimator $\widetilde{Q}_0$

for $k = 0, 1, 2, \ldots, K-1$ **do**

Sample i.i.d. observations $\{(S_i, A_i), i \in [n]\}$ from ν obtain $R_i \sim R(S_i, A_i)$ and $S_i' \sim P(S_i, A_i)$

Let $Y_i = R_i + \gamma \cdot \max_{a \in \mathcal{A}} \widetilde{Q}_k(S_i', a)$

Update action-value function:

$$\widetilde{Q}_{k+1} \leftarrow \underset{f \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} (Y_i - f(S_i, A_i))^2$$

Define π_K as the greedy policy w.r.t. $\widetilde{Q}_K$

Output: An estimator $\widetilde{Q}_K$ of Q^* and policy π_K

The theorem

For any $K \in \mathbb{N}$ let Q^{π_K} be the action-value function corresponding to policy π_K which is return by Algorithm 1, when run with a sparse ReLU network on the form

$$\mathcal{F}_0 = \{f(\cdot, a) \in \mathcal{F}(L^*, \{d_j^*\}_{j=0}^{L^*+1}, s^*) \mid a \in \mathcal{A}\}$$

where

$$L^* \lesssim (\log n)^{\xi'}, d_0 = r, d_j^*, d_{L+1} = 1, \lesssim n^{\xi'}, s^* \asymp n^{\alpha^*} \cdot (\log n)^{\xi'}$$

Let μ be any distribution over $\mathcal{S} \times \mathcal{A}$. Under *some assumptions*

$$\|Q^* - Q^{\pi_K}\|_{1,\mu} \leq C \cdot \frac{\phi_{\mu,\nu} \cdot \gamma}{(1-\gamma)^2} \cdot |\mathcal{A}| \cdot (\log n)^{\xi^*} \cdot n^{(\alpha^*-1)/2} + \frac{4\gamma^{K+1}}{(1-\gamma)^2} \cdot R_{\max}$$

Here $\alpha^* \in (0,1), C, \xi', \xi^*, \phi_{\mu,\nu} \in \mathbb{R}_+$ are constants depending on the assumptions and $R_{\max}$ the maximum possible reward.

Naive interpretation

This paper shows that for *any* MDP satisfying mild(?) assumptions, its optimal policy π^* can be approximated arbitrarily close by the fitted Q-iteration algorithm (in $\|\cdot\|_1$ over the chosen μ distribution). I.e. this algorithm 'solves' (up to any precision) a large class of games, decision processes, etc.

Problems

- Are the assumptions actually mild?
- How large are the constants / how quick is convergence?
- In the algorithm we assumed that we could solve a least squares problem on a function class of Neural Networks with several restrictions. According to one reviewer this is an NP-hard problem.
- Our result depended on a distribution μ, so it does not say much about how close we are to π^* outside the support of μ.
- The fitted Q-iteration algorithm differs from normal Deep Q-Learning in two important ways:
  - It avoids analysing errors in SGD and Back-Propagation by assuming that a global optimum is found.
  - It uses a fixed distribution on $\mathcal{S} \times A$ for batch sampling during *experience replay* rather than picking uniformly from actual experiences.

Hölder Smoothness

Definition

Let $\mathcal{D} \subseteq \mathbb{R}^r$ be compact and $\beta, H > 0$. A function $f : \mathcal{D} \to \mathbb{R}$ we call Hölder smooth if

$$\sum_{\alpha : |\alpha| < \beta} \|\partial^\alpha f\|_\infty + \sum_{\alpha : \|\alpha\|_1 = \lfloor \beta \rfloor} \sup_{x \neq y} \frac{|\partial^\alpha (f(x) - f(y))|}{\|x - y\|_\infty^{\beta - \lfloor \beta \rfloor}} \leq H$$

Where $\alpha = (\alpha_1, \dots, \alpha_r) \in \mathbb{N}^r$. We write $f \in C_r(\mathcal{D}, \beta, H)$.

We consider families of *Compositions of Hölder Functions*

$$\mathcal{G}(\{p_j, t_j, \beta_j, H_j\}_{j \in [q]})$$

where $t_j, p_j \in \mathbb{N}$, $t_j \leq p_j$ and $H_j, \beta_j > 0$, defined as containing f when $f = g_q \circ \cdots \circ g_1$ for $g_j : [a_j, b_j]^{p_j} \to [a_{j+1}, b_{j+1}]^{p_{j+1}}$ functions on some real hypercubes that only depend on t_j of their inputs for each of their components g_{jk}, and satisfies $g_{jk} \in C_{t_j}([a_j, b_j]_j^t, \beta_j, H_j)$.

Assumption 1, Hölder smoothness of $\mathcal{F}_0$ under T

Let

$$\mathcal{G}_0 = \{f : \mathcal{S} \times \mathcal{A} \to \mathbb{R} : f(\cdot, a) \in \mathcal{G}(\{p_j, t_j, \beta_j, H_j\}_{j\in[q]}), \forall a \in \mathcal{A}\}$$

It is assused that $Tf \in \mathcal{G}_0$ for any $f \in \mathcal{F}_0$.
I.e. when using the Bellman optimality operator on our sparse ReLU networks, we should stay in the class of compositions of Hölder smooth functions.

Assumption 2, Concentration Coefficients

Let $\nu_1, \nu_2 \in \mathcal{P}(\mathcal{S} \times \mathcal{A})$ be probability measures, Lebesgue-absolutely continuous in $\mathcal{S}$ Define

$$\kappa(m, \nu_1, \nu_2) = \sup_{\pi_1, \ldots, \pi_m} \left[\mathbb{E}_{\nu_2} \left(\frac{\mathrm{d}(P^{\pi_m} \ldots P^{\pi_1} \nu_1)}{\mathrm{d}\nu_2} \right)^2 \right]^{1/2}$$

Let ν be the sampling distribution from the algorithm, and *mu* the distribution over which we measure the error in the main theorem, then we assume

$$(1-\gamma)^2 \sum_{m \geq 1} \gamma^{m-1} m \kappa(m, \mu, \nu) = \phi_{\mu,\nu} < \infty$$