RL Essential

总体框架

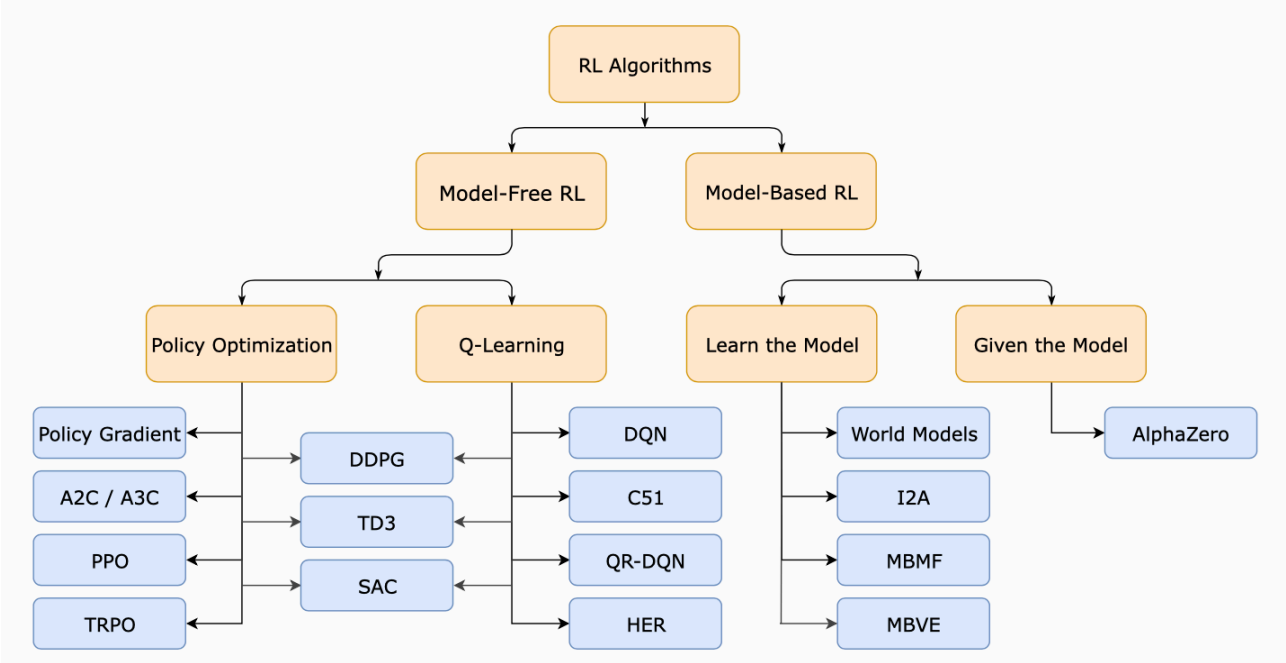
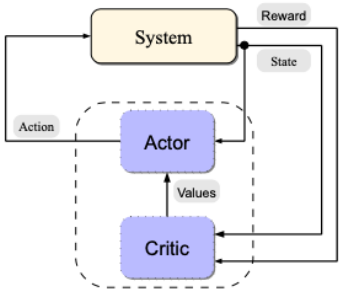
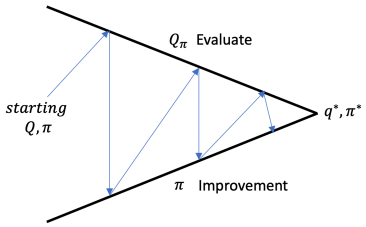
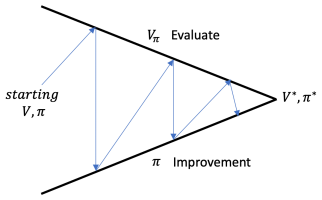


图 1

Actor-Critic







1. 使用什么方式计算 critic
   - MC
   - TD(0)
   - TD(n)
   - TD(λ)
2. 使用什么方式计算 actor
   - Greedy
   - Policy gradient
     - on-policy

  - off-policy
  - natural gradient

3. Behavior policy 与 target policy 匹配吗

   - on-policy
   - off-policy
   - Importance sampling
   - offline

基本概念

Defination

1. Policy

$$a_t \sim \pi_\theta(\cdot|s_t) \tag{1}$$

2. reward and return

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t \tag{2}$$

3. RL problem

$$P(\tau|\pi) = \rho_0(s_0) \prod_{t=0}^{T-1} P(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \tag{3}$$

$$\begin{aligned} J(\pi) &= \int_\tau P(\tau|\pi)R(\tau) \\ &= E_{\tau\sim\pi}[R(\tau)] \end{aligned} \tag{4}$$

找到一个最佳的 policy 使得累计 reward 最大化

$$\pi^* = \arg\max_\pi J(\pi) \tag{5}$$

Value Function

1. Value function

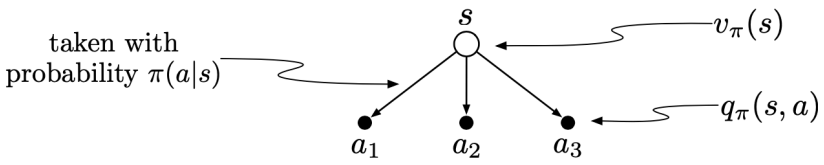
$$V^\pi(s) = E_{\tau\sim\pi}[R(\tau)\,|s_0 = s] \tag{6}$$

$$Q^\pi(s,a) = E_{\tau\sim\pi}[R(\tau)\,|s_0 = s, a_0 = a] \tag{7}$$

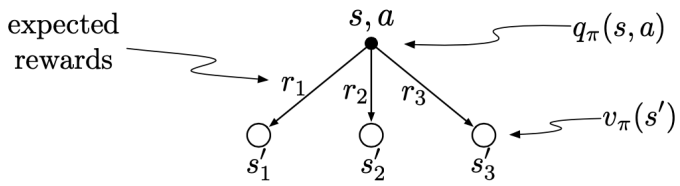
2. one-step expand for value function

$$V^\pi(s) = E_{a\sim\pi}[Q^\pi(s,a)] \tag{8}$$

$$Q^\pi(s,a) = r(s,a) + \gamma E_{s'\sim P}[V^\pi(s')] \tag{9}$$



(a) $V(s)$ 单步展开



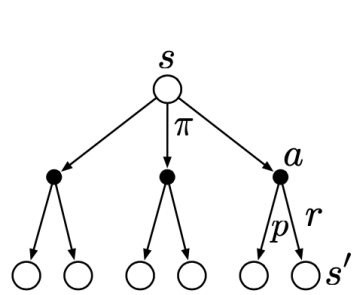
(b) $Q(s,a)$ 单步展开

3. Bellman equation (two-step expand for value function)

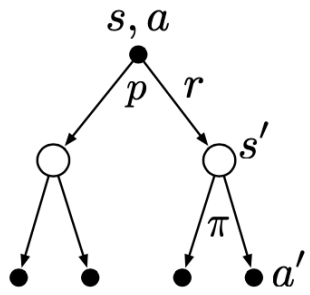
$$V^\pi(s) = E_{a\sim\pi s'\sim P}[r(s,a) + \gamma V^\pi(s')] \tag{10}$$

$$Q^\pi(s,a) = E_{s'\sim P}[r(s,a) + \gamma E_{a'\sim\pi}[Q^\pi(s',a')] \tag{11}$$

$$= r(s,a) + \gamma E_{a'\sim\pi s'\sim P}[Q^\pi(s',a')] \tag{12}$$



Backup diagram for v_π



q_π backup diagram

(a) $V(s)$ 两步展开

(b) $Q(s,a)$ 两步展开

Optimal Value Function

1. Optimal value function

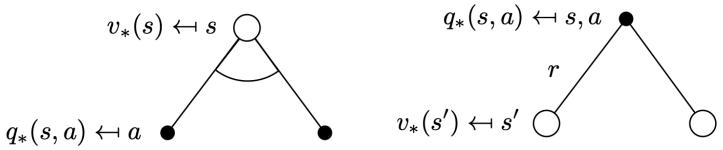
$$V^*(s) = \max_\pi V_\pi(s) = \max_\pi E_{\tau\sim\pi}[R(\tau)\,|s_0 = s] \tag{13}$$

$$Q^*(s,a) = \max_\pi Q_\pi(s,a) = \max_\pi E_{\tau\sim\pi}[R(\tau)\,|s_0 = s, a_0 = a] \tag{14}$$

2. one-step expand for optimal value function

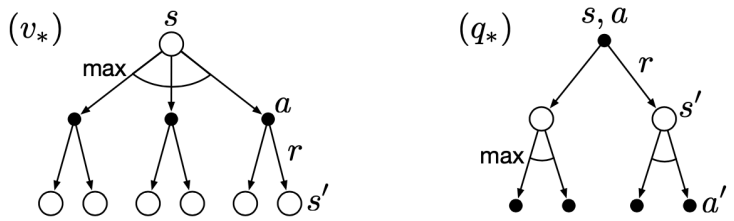
$$V^*(s) = \max_a Q^*(s,a) \tag{15}$$

$$Q^*(s,a) = r(s,a) + E_{s'\sim P}[V^*(s')] \tag{16}$$



(a) $V^*(s)$ 单步展开　　(b) $Q^*(s,a)$ 单步展开

3. Bellman equation for optimal value function((two-step expand)



$$V^*(s) = \max_a E_{s' \sim P}[r(s,a) + \gamma V^*(s')] \tag{17}$$

$$Q^*(s,a) = E_{s' \sim P}[r(s,a) + \gamma \max_{a'} Q^*(s',a')] \tag{18}$$

Advantage

Advantage 在 policy gradient 的算法中经常使用，此处给出定义，直觉上该值表征了在状态 s_t 选择特定动作的收益相对于平均收益的优势，另外可以证明 **TD-error 是 advantage 的无偏估计**

$$\begin{aligned} A_t &= R(\tau|s,a) - \mathsf{E}_{a\sim\pi}[R(\tau|s,a)] && (19)\\ &= Q(s_t,a_t) - V(s_t) && (20)\\ &= R(\tau) - V(s_t) && (21) \end{aligned}$$

这个性质可以极大的拓展 advantage 的估计方法，使之可以嫁接到几乎任意值函数的估计过程中，同样的也就存在 $A^{(1)}, A^{(2)}..., A^{(n)}$ 以及 $A^{(\lambda)}$,advantage 的估计几乎总是和值函数的估计紧密相连的。

$$\begin{aligned} \mathsf{E}_\pi[\delta^\pi|s_t,a_t] &= \mathsf{E}_\pi[r_t + \gamma V(s_{t+1})|s_t,a_t] - V(s_t)\\ &= Q(s_t,a_t) - V(s_t)\\ &= A(s_t,a_t) \end{aligned} \tag{22}$$

Implementing a critic

对一个未知的 MDP 过程，策略已知的情况下如何评价策略的优劣就是 critic 要解决的问题，从最朴素的逻辑看来，就是将该策略在环境中跑上很多次，甚至无限次，将其总收益的均值作为真实回报的估计值，这样的样本利用率无疑是很低的，需要采样很多次才能收敛于一个稳定的估计，但可以证明这种朴素的算法得到的估计是无偏的，其实这正是大名鼎鼎的蒙特卡洛算法。

另一种想法是没必要每次都跑到最后，如果已经遇到了之前碰到的情况就没必要再继续实验了，举例来说，馋糖-1，偷吃糖 +5，被妈妈打-10，考虑策略分布不变的前提下，如果开始馋糖，那不继续步进下去也可以计算最后的收益，换言之只需要在环境中采样一段数据的回报再加上记住的回报就是最终估计的总回报，这便是 TD 算法，和动态规划不同的无非是通过采样进行。

Monte-Carlo

1. On-policy

$$V_\pi = E_\pi[r_{t+1} + \gamma r_{t+2} + ... + \gamma^{T-1} r_T | S_t = s] \tag{23}$$

使用待评估策略生成的数据，取其平均作为对策略回报的估计

2. Off-policy Prediction via Importance Sampling

要区分 on-policy 和 off-policy 就首先要区分两类策略，一类是用来生成数据的叫做 behavior policy, 这个策略可以是千奇百怪的，甚至是完全随机的，而另一类叫 target policy, 这是我们心心念念要去学习的策略，而如果这两种策略不是一致的，则变成了 off-policy，让 actor 根据别人做的动作去学习，是需要矫正的，这也就是 Importance Sampling 的作用。根据另一个分布中的采样数据来估计当前分布的期望。

$$\mathsf{E}_{x\sim p(x)}[f(x)] = \int p(x) f(x) dx \tag{24}$$

$$= \int \frac{p(x)}{q(x)} q(x) f(x) dx \tag{25}$$

$$= \mathsf{E}_{x\sim q(x)}[\frac{p(x)}{q(x)} f(x)] \tag{26}$$

trajectory 的分布也会随着 target policy 与 behavior policy 的不同产生差异，importance sampling ratio 是一个重要指标，来刻画两个 tarjectory 分布的比值，有助于化简之后的公式，在论文中也经常出现

$$\rho_{0:T} = \frac{P(\tau|\pi_{target})}{P(\tau|\pi_{behavior})} \tag{27}$$

$$= \frac{\rho_0(s_0) \prod_{t=0}^{T-1} P(s_{t+1}|s_t, a_t) \pi_{target}(a_t|s_t)}{\rho_0(s_0) \prod_{t=0}^{T-1} P(s_{t+1}|s_t, a_t) \pi_{behavior}(a_t|s_t)} \tag{28}$$

$$= \prod_{t=0}^{T-1} \frac{\pi_{target}(a_t|s_t)}{\pi_{behavior}(a_t|s_t)} \tag{29}$$

we wish to estimate the expected returns (values) under the target policy, but all we have are returns G_t due to the behavior policy. 我们应该估计的是使用 target policy 做采样时的回报期望，但是只有使用 behavior policy 采样后得到的 G_t :

$$\mathsf{E}[G_t|s_0 = s_t] = v_b(s_t) \tag{30}$$

此时如果想要评估 target policy 产生的 $v_\pi(s_t)$ 就需要使用 Importance Sampling（26）此时采样的数据分布是由 $\pi_{behavior}$ 产生的，而期望评估的是 π_{target} 对应的值函数

$$\mathsf{E}[\rho_{t:T-1}G_T|s_0 = s_t] = v_\pi(s_t) \tag{31}$$

Temporal-Difference

TD(0)

1. TD target

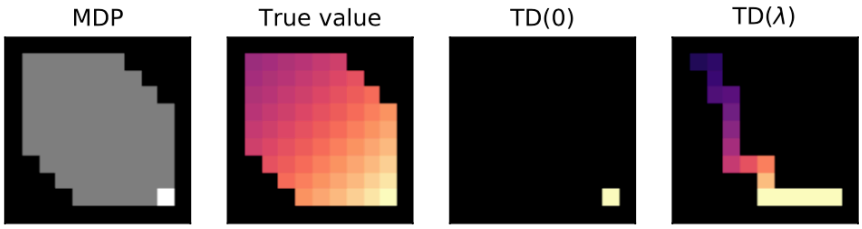
$$G_{t:t+1} = r_t + \gamma V(S_{t+1}) \tag{32}$$

2. TD error

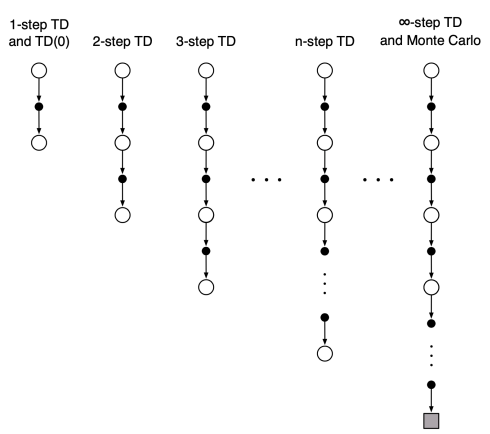
$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t) \tag{33}$$

3. Connection between MC

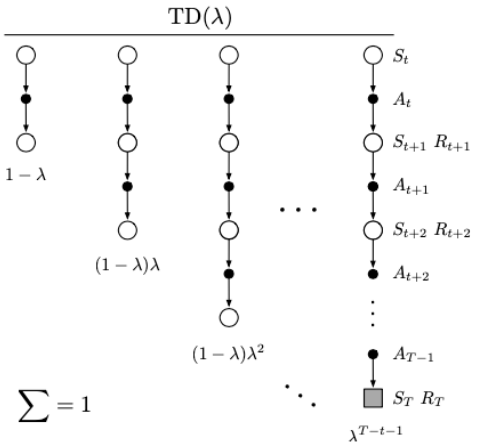
MC error 可以被视为一系列 TD error 的累加，也就是 MC 方法在更新价值函数时，相当于考虑了每一步的 TD error，而 TD(0) 只考虑当下的 error 从而更新价值函数，这就涉及到了 credit assignment problem，当获得 reward 信号后，如何将有关联的值函数更新，例如考试没通过，到底是和昨晚上通宵复习临阵磨枪相关呢，还是和平时划水相关呢，TD(0) 只会更新当下的值函数，v(通宵学习) = -100 + v(考试结束) 显然有些荒谬，换言之 TD(0) 对信息的传播是缓慢的，拿到 reward 信号后，当下的 state 的值函数可以得到更新，但是更早的状态却没有更新。下图是一个 MDP 过程，从左上角出发，agent 做随机动作直到右下角可以得到 reward，可以看出各个不同方法的区别。



$$\begin{aligned}
G_t - V(S_t) &= R_{t+1} + \gamma G_{t+1} - V(S_t) + \gamma V(S_{t+1}) - \gamma V(S_{t+1}) \\
&= \delta_t + \gamma (G_{t+1} - V(S_{t+1})) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2 (G_{t+2} - V(S_{t+2})) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}(G_T - V(S_T)) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}(0-0) \\
&= \sum_{k=t}^{T-1} \gamma^{k-t}\delta_k
\end{aligned} \tag{34}$$



(a) $TD(n)$



(b) $TD(\lambda)$

TD(n)

1. TD target

$$G_{t:t+n} = r_t + \gamma r_{t+1} + ... + \gamma^{n-1} r_{t+n-1} + \gamma^n V(s_{t+n}) \tag{35}$$

2. TD error

$$\delta = G_{t:t+n} - V(s_t) \tag{36}$$

TD(λ)

1. λ return

   MC 和 TD(0) 代表了两个极端, 可以在这两者之间做一个类似差值的工作，从而平衡两种方法。

$$G_t = r_t + G_{t+1} \tag{37}$$

$$G_t = r_t + V(S_{t+1}) \tag{38}$$

$$G^{\lambda}_{t:T} = r_t + \gamma \left[(1-\lambda) V(S_{t+1}) + \lambda G^{\lambda}_{t+1:T} \right] \tag{39}$$

2. TD target

$$G^{\lambda}_t = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n} \tag{40}$$

$$= (1-\lambda) \sum_{n=1}^{T-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t-1} G_t \tag{41}$$

3. TD error

$$\delta^{\lambda} = G^{\lambda} - V(s_t) \tag{42}$$

从公式（41）可以看出 $TD(\lambda$ 与 MC，TD 的联系，当 $\lambda = 1$ 时 G^{λ}_t 退化为 G_t 也就是蒙特卡洛，而当 $\lambda = 0$ 时 $G^{\lambda}_t = G_{t:t+1}$ 也就是 $TD(0)$ 一方面是无偏估计但是高方差，一方面是较低的方差但是估计有偏，所谓的 bias-variance balance problem 会经常出现，强化学习中有三对很重要的权衡: (1) bias-variance (2)

Semi-gradient TD(λ) for estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$
Algorithm parameters: step size $\alpha > 0$, trace decay rate $\lambda \in [0, 1]$
Initialize value-function weights $\mathbf{w}$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
  Initialize S
  $\mathbf{z} \leftarrow \mathbf{0}$ (a d-dimensional vector)
  Loop for each step of episode:
  | Choose $A \sim \pi(\cdot|S)$
  | Take action A, observe R, S'
  | $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z} + \nabla\hat{v}(S,\mathbf{w})$
  | $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$
  | $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
  | $S \leftarrow S'$
  until S' is terminal

图 6: Eligibility Traces

exploration-exploitation (3) on policy - off policy 往往交接处就是创新的涌现点，这三方面都有很多精彩的工作试图鱼与熊掌兼得，或是更好的权衡两者找到平衡点。

$TD(\lambda)$ 相当于对不同步长进行了加权，这就使得更新只能在环境运行结束后进行，而可以实时更新价值函数恰恰是 TD 方法最大的优点，为了能在环境运行时同步的计算 $TD(\lambda)$ 需要引进"backward view using eligibility traces" Backward 指的是,考虑过去价值对当前价值的影响。简而言之维持了一个 eligibility traces weight 和一个 long term weight 将梯度不断的累加到 eligibility traces weight 中但同时以 $\gamma\lambda$ 的速率衰减，正如人会逐步遗忘过去的经验，更新 $v_\omega(s)$ 时则同时考虑两者。

$$\begin{aligned} \mathbf{z}_{-1} &\doteq \mathbf{0} \\ \mathbf{z}_t &\doteq \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}\left(S_t, \mathbf{w}_t\right), \quad 0 \leq t \leq T \end{aligned} \tag{43}$$

单步的 TD-error 为 $\delta_t \doteq R_{t+1} + \gamma\hat{v}\left(S_{t+1}, \mathbf{w}_t\right) - \hat{v}\left(S_t, \mathbf{w}_t\right)$ 因此更新价值函数的公式即为:

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t\mathbf{z}_t \tag{44}$$

对比一般的价值函数更新公式：$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t\nabla\hat{v}\left(S_t, \mathbf{w}_t\right)$ 最大的差异在于 $TD(\lambda)$ 相当于对之前各个时刻的梯度做了加权，后者只考虑了此刻的梯度。

GAE

$$\begin{aligned} \hat{A}_t^{(1)} &:= \delta_t^V = -V\left(s_t\right) + r_t + \gamma V\left(s_{t+1}\right) \\ \hat{A}_t^{(2)} &:= \delta_t^V + \gamma\delta_{t+1}^V = -V\left(s_t\right) + r_t + \gamma r_{t+1} + \gamma^2 V\left(s_{t+2}\right) \\ \hat{A}_t^{(3)} &:= \delta_t^V + \gamma\delta_{t+1}^V + \gamma^2\delta_{t+2}^V = -V\left(s_t\right) + r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 V\left(s_{t+3}\right) \\ \hat{A}_t^{(k)} &:= \sum_{l=0}^{k-1}\gamma^l\delta_{t+l}^V = -V\left(s_t\right) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V\left(s_{t+k}\right) \end{aligned} \tag{45}$$

仿照 $TD(\lambda)$ 形式，$GAE(\lambda, \gamma)$ 可以定义为一个对 advantage 的 exponentially-weighted-average, 从而降低估计的方差。

$$
\begin{aligned}
\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} &:= (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + ...\right) \\
&= (1-\lambda)\left(\delta_t^V + \lambda\left(\delta_t^V + \gamma\delta_{t+1}^V\right) + \lambda^2\left(\delta_t^V + \gamma\delta_{t+1}^V + \gamma^2\delta_{t+2}^V\right) + ...\right) \\
&= (1-\lambda)\left(\delta_t^V\left(1 + \lambda + \lambda^2 + ...\right) + \gamma\delta_{t+1}^V\left(\lambda + \lambda^2 + \lambda^3 + ...\right)\right. \\
&\quad \left.+\gamma^2\delta_{t+2}^V\left(\lambda^2 + \lambda^3 + \lambda^4 + ...\right) + ...\right) \\
&= (1-\lambda)\left(\delta_t^V\left(\frac{1}{1-\lambda}\right) + \gamma\delta_{t+1}^V\left(\frac{\lambda}{1-\lambda}\right) + \gamma^2\delta_{t+2}^V\left(\frac{\lambda^2}{1-\lambda}\right) + ...\right) \\
&= \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}^V
\end{aligned}
\tag{46}
$$

可以发现在 $\lambda = 0$ 时，退化为 TD(0)，$\lambda = 1$ 时退化为蒙特卡洛

$$
\mathrm{GAE}(\gamma, 0) : \hat{A}_t := \delta_t \quad = r_t + \gamma V\left(s_{t+1}\right) - V\left(s_t\right) \tag{47}
$$

$$
\mathrm{GAE}(\gamma, 1) : \hat{A}_t := \sum_{l=0}^{\infty}\gamma^l\delta_{t+l} = \sum_{l=0}^{\infty}\gamma^l r_{t+l} - V\left(s_t\right) \tag{48}
$$

GAE 与 $TD(\lambda)$ 的关系是清晰的，GAE 就是 $\delta^\lambda = G_t^\lambda - V(s_t)$ 的另一种计算形式，但是采用 GAE 更适用于对 advantage 的估计:

$$
\widehat{A^\lambda}\left(s_t, a_t; \omega\right) := (1-\lambda)\sum_{i=t}^{T-1}\lambda^{i-t}\left(R_{t:i+1} - \widehat{V}\left(s_t; \omega\right)\right) + \lambda^{T-t}\left(R_{t:T+1} - \widehat{V}\left(s_t; \omega\right)\right) \tag{49}
$$

$$
= (1-\lambda)\sum_{i=t}^{T-1}\lambda^{i-t}R_{t:i+1} + \lambda^{T-t}R_{t:T+1} - \widehat{V}\left(s_t; \omega\right) \tag{50}
$$

$$
= R_t^\lambda - \widehat{V}\left(s_t; \omega\right) \tag{51}
$$

$$
= \delta_V^\lambda\left(s_t, s_{t+1}; \omega\right) \tag{52}
$$

Implementing a actor

策略提升无非有两种方式，一种是根据现有的值函数贪婪的选择动作，另一种是直接对强化学习的优化目标进行策略提升，但其实几乎所有工程实践中使用的方式都可以放入 actor-critic 的框架，greedy 策略可以视为一种特殊的 actor，而 policy gradient 策略无非是使用了更为复杂的 actor

Greedy

$$
\pi(a|s) = \underset{a\in\mathcal{A}}{\mathrm{argmax}}\{r(s,a) + \mathcal{P}(s'|s,a)V\left(s'\right)\} \tag{53}
$$

$$
= \underset{a\in\mathcal{A}}{\mathrm{argmax}}Q(s,a) \tag{54}
$$

如果已知 $V(s_t)$ 则可以贪婪的选择下一时刻的动作，但此时要求 MDP 过程的状态转移函数为已知条件，因此在实际使用时几乎都会去计算 $Q(s_t, a_t)$ 或 $Q^*(s_t, a_t)$ 从而很方便的得到最优化动作, 整体学习过程就是值函数评估和策略提升的迭代过程:

$$
\pi_0 \xrightarrow{\mathrm{E}} q_{\pi_0} \xrightarrow{\mathrm{I}} \pi_1 \xrightarrow{\mathrm{E}} q_{\pi_1} \xrightarrow{\mathrm{I}} \pi_2 \xrightarrow{\mathrm{E}} \cdots \xrightarrow{\mathrm{I}} \pi_* \xrightarrow{\mathrm{E}} q_*
$$

值得注意此时的算法都是以 on-policy 的形式进行的, 因为评估的值函数必须要对应生成数据的 policy 也就是 target policy 与 behavior policy 是一致的，如果希望使用离线数据，则必须进行修正。有没有一种方式可以使用离线数据但又不需要 IS 呢? 注意到最优值函数的 bellam equation (18) 这里排除了对策略的依赖，因为每次更新时都使用了当前最佳策略生成的动作，因此能以 off-policy 的方式进行学习并且不需要使用 IS 进行矫正, 换言之只要学习到了 $Q^*(s,a)$ 就可以还原出最优化策略和动作，如下:

$$\pi(a|s) = \underset{a\in\mathcal{A}}{\text{argmax}} Q^*(s,a) \tag{55}$$

$\epsilon-$greedy 对所有动作都赋予一个非零概率，再从其中采样，有 ϵ 对概率使用随机动作，有 $1-\epsilon$ 的概率使用 $a^* = \text{argmax}_{a'\in\mathcal{A}}(Q(s,a'))$ 换言之以一定的概率使用随机动作，以此来增强探索。

$$\pi(a \mid s_t) \leftarrow \begin{cases} 1-\varepsilon+\varepsilon/\left|\mathcal{A}\right| & \text{if } a = \text{argmax}_{a'\in\mathcal{A}}(Q(s,a')) \\ \varepsilon/\left|\mathcal{A}\right| & \text{otherwise} \end{cases} \tag{56}$$

Boltzmann distribution

$$\mathsf{Pr}\{A_t = a\} = \frac{e^{Q_t(a)/\tau}}{\sum_{b=1}^{n} e^{Q_t(b)/\tau}} \tag{57}$$

$\epsilon-greedy$ 策略随机的选择动作但这也意味着对于好的动作和差的动作，被选择的概率都相同，一种改进是对不同动作赋予不同的概率，再从这个分布中进行采样。τ 是控制动作离散程度的超参数，物理含义是温度，τ 越大则越接近于均匀分布，反之则接近于值函数的分布，当 $\tau\to 0$ 时，退化为 greedy 策略。

Policy-gradient on-policy 另一种思路是用一种策略去直接的最大化回报，而非隐式的从值函数中还原出策略，回忆强化学习的根本问题，最大化累计回报，以此作为优化目标直接的更新策略:

$$J(\pi_\theta) = \mathsf{E}_{\tau\sim\pi_\theta}[R(\tau)] \tag{58}$$

$$= \mathsf{E}_{\substack{\tau\sim\pi_\theta \\ s\sim\mathcal{S}}}[R(\tau)|s_0 = s] \tag{59}$$

$$= \mathsf{E}_{s\sim\mathcal{S}}[V^\pi(s)] \tag{60}$$

有了优化目标，最朴素的做法就是求一下梯度，做梯度下降去优化该函数:

$$\theta_{k+1} = \theta_k + \alpha\nabla_\theta J(\pi_\theta) \tag{61}$$

回忆 tarjectory 的分布, 其中的环境状态转移函数是未知的:

$$\pi(\tau|\theta) = P(\tau|\theta) = P_\theta(s1, a1, ..., s_T, a_T) = \rho_0(s_0)\prod_{t=0}^{T} P(s_{t+1}|s_t, a_t)\pi_\theta(a_t|s_t) \tag{62}$$

先求一下累计回报的相对于策略的梯度看看是否奏效:

$$\nabla J(\pi_\theta) = \nabla_\theta \mathsf{E}_{\tau\sim\pi_\theta}[R(\tau)] \tag{63}$$

$$= \nabla_\theta \int_\tau P(\tau|\theta)R(\tau) \qquad \text{Expand expectation} \tag{64}$$

$$= \int_\tau \nabla_\theta P(\tau|\theta)R(\tau) \qquad \text{Bring gradient under integral} \tag{65}$$

$$= \int_\tau P(\tau|\theta)\nabla_\theta \log(P(\tau|\theta))R(\tau) \qquad \text{Log-derivative trick} \tag{66}$$

$$= \mathsf{E}_{\tau\sim\pi_\theta}[\nabla_\theta \log(P(\tau|\theta))R(\tau)] \qquad \text{Return to expectation form} \tag{67}$$

注意到（67）中的 $\nabla_\theta \log(P(\tau|\theta))$ 这是依赖于环境转移函数的（62）所以需要进一步处理，在求梯度的过程中不包含 θ 的项自然会是零，这就排除了对状态转移函数的需求:

$$\nabla_\theta \log P(\tau|\theta) = \nabla_\theta \log \rho_0(s_0) + \sum_{t=0}^{T} \Big(\nabla_\theta \log P(s_{t+1}|s_t, a_t) + \nabla_\theta \log \pi_\theta(a_t|s_t) \Big) \tag{68}$$

$$= \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \tag{69}$$

结合（67）和（69）可以得出核心公式:

$$\nabla_\theta J(\pi_\theta) = \mathsf{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)] \tag{70}$$

可以进一步得到更为一般化的推导:

$$\nabla_\theta J(\pi_\theta) = \mathsf{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t}^{T} \gamma^{t-t'} r_t] \tag{71}$$

$$= \mathsf{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \mathsf{E}_{\tau\sim\pi_\theta}[R_t|s_t, a_t]] \tag{72}$$

$$= \mathsf{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) Q(s_t, a_t)] \tag{73}$$

此时就可以利用样本来估计该期望，agent 使用 π_θ 与环境进行交互，收集数据 $\mathcal{D} = \{\tau_i\}_{i=1,\dots,N}$ 去估计策略梯度，注意此时的 target policy 和 behavior policy 需要是一致的，也就是 on-policy 的方式训练，最直觉的例子就是使用蒙特卡洛去估计 Q-function，换言之就是直接跑到环境结束为止，然后把在此期间收集到的 reward 加起来，取平均来估计 $Q(s_t, a_t)$

$$\hat{g} = \frac{1}{|\mathcal{D}|} \sum_{\tau\in\mathcal{D}} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) G_t \tag{74}$$

$$= \frac{1}{|\mathcal{D}|} \sum_{\tau\in\mathcal{D}} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{k=t}^{T} \gamma^{k-t} r_t \qquad \text{reward to go} \tag{75}$$

此时的估计虽然是无偏的但是方差较大，为了减小方差可以考虑使用 advantage 去替换 Q 此时的估计依旧是无偏的，先给出一个有用的等式:

$$\mathsf{E}_{x\sim P_\theta}[\nabla_\theta \log P_\theta(x)] = \int P_\theta(x) \nabla_\theta \log P_\theta(x) dx \tag{76}$$

$$= \int \nabla P_\theta(x) dx \tag{77}$$

$$= \nabla \int P_\theta(x) dx \tag{78}$$

$$= 0 \tag{79}$$

进一步就可以得到如下等式:

$$\mathsf{E}_{a_t\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(a_t|s_t) b(s_t)] = 0 \tag{80}$$

$$\nabla_\theta J(\pi_\theta) = \mathsf{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1}) - b(s_t)\right)\right] \tag{81}$$

最常见的形式是 $b(s_t) = V(s_t)$ 此时 $A(s_t) = Q(s_t, a_t) - V(s_t)$ 参考 (21)(**??**) 在常见的工程实践中，最常用的是将更新 $V(s_t)$ 时的 TD-error 作为 advantage 的估计，在此基础上有一系列等价的形式去估计该策略梯度，也因此演化为几种不同的算法:

$$\begin{aligned}
\nabla_\theta J(\theta) &= \mathsf{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(A \mid S) G_t\right] && \text{REINFORCE} && (82)\\
&= \mathsf{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(A \mid S) q_w(S, A)\right] && \text{Q Actor-Critic} && (83)\\
&= \mathsf{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(A \mid S) A_w(S, A)\right] && \text{Advantage Actor-Critic} && (84)\\
&= \mathsf{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(A \mid S) \delta_t\right] && \text{TD Actor-Critic} && (85)\\
&= \mathsf{E}_{\pi_\theta}\left[\delta_t e_t\right] && \text{TD}(\lambda)\text{Actor-Critic} && (86)
\end{aligned}$$

Policy-gradient off-policy 从最抽象的层次说，此问题相当于要使用 behavior policy 生成的样本序列 $\tau \sim \pi_{\theta^b}$ 去对 target policy 的优化目标 $J(\pi_\theta)$ 进行优化，对于 π_θ 的梯度是不受影响的，主要难点在于如何估计此时的 advantage，此时 Importance Sampling 就可以大展身手了，公式（31）已经给出了离线估计值函数的方法，而只要将其带入 (70) 即可:

$$\nabla_\theta J(\pi_\theta) = \mathsf{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{T}\rho_{t:T}\nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)] \tag{87}$$

$$\nabla_\theta J(\pi_\theta) = \mathsf{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\prod_{t'=t}^{T-1}\frac{\pi_{target}(a'_t|s'_t)}{\pi_{behavior}(a'_t|s'_t)}\nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)\right] \tag{88}$$

但这是几乎无法计算的，因为在每一个时刻都需要对未来所有时刻进行累乘，转化一下思路将其变为对动作的期望，去掉对动作序列的依赖，并做一些近似使问题可解，注意到（60）PG 的优化目标其实相当于在所有起始点都希望最终回报最大化，引入 $d^b(s) = \lim_{t\to\infty} P(s_t = s|s_0, b)$ 以此来描述执行 behavior-policy π_b 后 state 的分布:

$$J(\theta) = \mathsf{E}_{s\sim d^b}[V^{\pi_\theta}(s)]$$

回忆 V 与 Q 的关系（8）可以进一步得到:

$$\begin{aligned}
J(\theta) &= \mathsf{E}_{s\sim d^b}[\mathsf{E}_{a\sim\pi_\theta}[Q^{\pi_\theta}(s, a)]] && (89)\\
&= \mathsf{E}_{s\sim d^b}\left[\sum_{a\in\mathcal{A}} Q^{\pi_\theta}(s, a)\pi_\theta(a \mid s)\right] && (90)
\end{aligned}$$

以此作为基础，对于无法计算的项进行忽略（收敛性证明见原始论文）就可以得到适用于 Off-policy 形式的梯度估计。其利用了 IS 处理不同策略采样数据分布不同的影响，虽然略显繁琐但却是一系列算法的理论基础，例如 IMPALA 的 V-traces.

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \mathsf{E}_{s\sim d^b}\left[\sum_{a\in\mathcal{A}} Q^{\pi_\theta}(s,a)\pi_\theta(a\mid s)\right] \\
&= \mathsf{E}_{s\sim d^b}\left[\sum_{a\in\mathcal{A}} (Q^{\pi_\theta}(s,a)\nabla_\theta\pi_\theta(a\mid s) + \pi_\theta(a\mid s)\nabla_\theta Q^{\pi_\theta}(s,a))\right] \\
&\approx \mathsf{E}_{s\sim d^b}\left[\sum_{a\in\mathcal{A}} Q^{\pi_\theta}(s,a)\nabla_\theta\pi_\theta(a\mid s)\right] \\
&= \mathsf{E}_{s\sim d^b}\left[\sum_{a\in\mathcal{A}} \pi_b(a\mid s)\frac{\pi_\theta(a\mid s)}{\pi_b(a\mid s)} Q^{\pi_\theta}(s,a)\frac{\nabla_\theta\pi_\theta(a\mid s)}{\pi_\theta(a\mid s)}\right] \\
&= \mathsf{E}_{s\sim d^b}\left[\mathsf{E}_{a\sim\pi_b(a|s)}[\frac{\pi_\theta(a\mid s)}{\pi_b(a\mid s)} Q^{\pi_\theta}(s,a)\nabla_\theta \ln\pi_\theta(a\mid s)]\right] \\
&= \mathsf{E}_{\tau\sim\pi_b}\left[\frac{\pi_\theta(a\mid s)}{\pi_b(a\mid s)} Q^{\pi_\theta}(s,a)\nabla_\theta \log\pi_\theta(a\mid s)\right]
\end{aligned}
\tag{91}
$$

有了公式 (91) 后就可以采用任何喜欢的方式来构建算法了，例如使用 π_b 生成数据 $\mathcal{D}$ 后使用 TD(0) 更新 critic:

$$
\delta_t = \frac{\pi_\theta(a_t\mid s_t)}{\pi_b(a_t\mid s_t)}[r_t + v(s_{t+1}) - v_{s_t}] \tag{92}
$$

$$
\nabla_\theta J(\theta) = \frac{1}{|\mathcal{D}|}\sum_{\tau\sim\mathcal{D}}\sum_{t=0}^{T}[\delta_t \nabla_\theta \log\pi_\theta(a_t\mid s_t)] \tag{93}
$$

Entropy regularized framework

basic idea

为方便参考，给出熵与 KL 的计算公式如下：

$$
\begin{aligned}
H(\pi(a|s)) &= \mathsf{E}_{a\sim\pi(a|s)}[-\log(\pi(a|s))] &(94)\\
&= -\sum_a \pi(a|s)\log(\pi(a|s)) &(95)
\end{aligned}
$$

$$
\begin{aligned}
D_{\mathrm{KL}}(P\|Q) &= \int_{-\infty}^{\infty} p(x)\log\left(\frac{p(x)}{q(x)}\right)dx &(96)\\
&= \sum_{x\in\mathcal{X}} P(x)\log\left(\frac{P(x)}{Q(x)}\right) &(97)
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{H}(X) &= \mathrm{E}\left[\mathrm{I}_X(x)\right] &(98)\\
&= \log(N) - D_{\mathrm{KL}}\left(p_X(x)\|P_U(X)\right) &(99)
\end{aligned}
$$

从更统一的框架看，定义当前策略与参考策略在某时刻的 KL，如果想还原出原始的熵正则化，只需要取参考分布为均匀分布即可 $\bar{\pi}\sim U$.

$$
\mathrm{KL}_t = D_{\mathrm{KL}}\left[\pi\left(\cdot\mid s_t\right)\|\bar{\pi}\left(\cdot\mid s_t\right)\right] \tag{100}
$$

加入正则化项后的累计回报如下：

$$
\sum_{t=0}^{\infty}\gamma^t\left(r_t - \tau\mathrm{KL}_t\right) \tag{101}
$$

从而可以定义值函数:

$$V_{\pi}(s)=\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^{t}\left(r_{t}-\tau\mathrm{KL}_{t}\right)\mid s_{0}=s\right] \tag{102}$$

$$Q_{\pi}(s,a)=\mathbb{E}\left[r_{0}+\sum_{t=1}^{\infty}\gamma^{t}\left(r_{t}-\tau\mathrm{KL}_{t}\right)\mid s_{0}=s,a_{0}=a\right] \tag{103}$$

注意到 Q function 的 r_0 项并不包括 KL penality，因为 a_0 是选择的确定性动作，因此 Q 与 V 的关系如下：

$$V_{\pi}(s)=\mathbb{E}_{a\sim\pi}\left[Q_{\pi}(s,a)\right]-\tau\mathrm{KL}(s) \tag{104}$$

Continuous action spaces

Policy Function

$$\pi^{*}=\arg\max_{\pi}\sum_{t}\mathbb{E}_{(\mathbf{s}_{t},\mathbf{a}_{t})\sim\rho_{\pi}}\left[r\left(\mathbf{s}_{t},\mathbf{a}_{t}\right)+\alpha\mathcal{H}\left(\pi\left(\cdot\mid\mathbf{s}_{t}\right)\right)\right] \tag{105}$$

$$\pi_{\text{new}}=\arg\min_{\pi'\in\Pi}D_{\mathrm{KL}}\left(\pi'\left(.\mid s_{t}\right)\|\frac{\exp\left(\frac{1}{\alpha}Q^{\pi_{\text{old}}}\left(s_{t},.\right)\right)}{Z^{\pi_{\text{old}}}\left(s_{t}\right)}\right) \tag{106}$$

$$=\arg\min_{\pi'\in\Pi}D_{\mathrm{KL}}\left(\pi'\left(.\mid s_{t}\right)\|\exp\left(\frac{1}{\alpha}Q^{\pi_{\text{old}}}\left(s_{t},.\right)-\log Z^{\pi_{\text{old}}}\left(s_{t}\right)\right)\right) \tag{107}$$

$$J_{\pi}(\theta)=D_{\mathrm{KL}}\left(\pi_{\theta}\left(.\mid s_{t}\right)\|\exp\left(\frac{1}{\alpha}Q^{w}\left(s_{t},.\right)-\log Z_{w}\left(s_{t}\right)\right)\right) \tag{108}$$

$$=\mathbb{E}_{a_{t}\sim\pi}\left[-\log\left(\frac{\exp\left(\frac{1}{\alpha}Q^{w}\left(s_{t},a_{t}\right)-\log Z_{w}\left(s_{t}\right)\right)}{\pi_{\theta}\left(a_{t}\mid s_{t}\right)}\right)\right] \tag{109}$$

$$=\mathbb{E}_{a_{t}\sim\pi}\left[\log\pi_{\theta}\left(a_{t}\mid s_{t}\right)-\frac{1}{\alpha}Q^{w}\left(s_{t},a_{t}\right)+\log Z_{w}\left(s_{t}\right)\right] \tag{110}$$

$$J_{\pi}(\theta)\propto\mathbb{E}_{a_{t}\sim\pi}\left[\alpha\log\pi_{\theta}\left(a_{t}\mid s_{t}\right)-Q^{w}\left(s_{t},a_{t}\right)\right] \tag{111}$$

Value Function

$$V\left(\mathbf{s}_{t}\right)=\mathbb{E}_{\mathbf{a}_{t}\sim\pi}\left[Q\left(\mathbf{s}_{t},\mathbf{a}_{t}\right)-\alpha\log\pi\left(\mathbf{a}_{t}\mid\mathbf{s}_{t}\right)\right] \tag{112}$$

$$J_{Q}(\theta)=\mathbb{E}_{(\mathbf{s}_{t},\mathbf{a}_{t})\sim\mathcal{D}}\left[\frac{1}{2}\left(Q_{\theta}\left(\mathbf{s}_{t},\mathbf{a}_{t}\right)-\left(r\left(\mathbf{s}_{t},\mathbf{a}_{t}\right)+\gamma\mathbb{E}_{\mathbf{s}_{t+1}\sim p}\left[V_{\bar{\theta}}\left(\mathbf{s}_{t+1}\right)\right]\right)\right)^{2}\right] \tag{113}$$

$$J_{\pi}(\phi)=\mathbb{E}_{\mathbf{s}_{t}\sim\mathcal{D}}\left[\mathbb{E}_{\mathbf{a}_{t}\sim\pi_{\phi}}\left[\alpha\log\left(\pi_{\phi}\left(\mathbf{a}_{t}\mid\mathbf{s}_{t}\right)\right)-Q_{\theta}\left(\mathbf{s}_{t},\pi_{\phi}(\mathbf{a}_{t}\mid\mathbf{s}_{t})\right)\right]\right] \tag{114}$$

$$\alpha_{t}^{*}=\arg\min_{\alpha_{t}}\mathbb{E}_{\mathbf{a}_{t}\sim\pi_{t}^{*}}\left[-\alpha_{t}\log\pi_{t}^{*}\left(\mathbf{a}_{t}\mid\mathbf{s}_{t};\alpha_{t}\right)-\alpha_{t}\overline{\mathcal{H}}\right] \tag{115}$$

$$J(\alpha)=\mathbb{E}_{\mathbf{a}_{t}\sim\pi_{t}}\left[-\alpha\log\pi_{t}\left(\mathbf{a}_{t}\mid\mathbf{s}_{t}\right)-\alpha\overline{\mathcal{H}}\right] \tag{116}$$

Discrete action spaces

$$V\left(s_{t}\right):=\pi\left(s_{t}\right)^{T}\left[Q\left(s_{t}\right)-\alpha\log\pi\left(s_{t}\right)\right] \tag{117}$$

$$J(\alpha)=\pi_{t}\left(s_{t}\right)^{T}\left[-\alpha\log\left(\pi_{t}\left(s_{t}\right)\right)-\alpha\bar{H}\right] \tag{118}$$

$$J_{\pi}(\phi) = E_{s_t \sim D}\left[\pi_t\left(s_t\right)^T\left[\alpha \log \pi_{\phi}\left(s_t\right) - Q_{\theta}\left(s_t\right)\right]\right] \tag{119}$$

$$Q^{\pi}(s,a) = \mathop{\mathrm{E}}_{\substack{s' \sim P \\ a' \sim \pi}}\left[R\left(s,a,s'\right) + \gamma\left(Q^{\pi}\left(s',a'\right) + \alpha H\left(\pi\left(\cdot \mid s'\right)\right)\right)\right] \tag{120}$$

$$= \mathop{\mathrm{E}}_{\substack{s' \sim P \\ a' \sim \pi}}\left[R\left(s,a,s'\right) + \gamma\left(Q^{\pi}\left(s',a'\right) + \alpha_{a' \sim \pi(a' \mid s')}\left[-\log \pi\left(a' \mid s'\right)\right]\right)\right] \tag{121}$$

$$= \mathop{\mathrm{E}}_{\substack{s' \sim P \\ a' \sim \pi}}\left[R\left(s,a,s'\right) + \gamma\left(Q^{\pi}\left(s',a'\right) - \alpha \log \pi\left(a' \mid s'\right)\right)\right] \qquad \text{continuous action space} \tag{122}$$

$$= \mathop{\mathrm{E}}_{\substack{s' \sim P \\ a'}}\left[R\left(s,a,s'\right) + \gamma\left(Q^{\pi}\left(s',a'\right) - \alpha \sum_{a' \sim \pi(a' \mid s')} \pi\left(a' \mid s'\right) \log \pi\left(a' \mid s'\right)\right)\right] \tag{123}$$

$$= \mathop{\mathrm{E}}_{s' \sim P}\left[R\left(s,a,s'\right) + \gamma \sum_{a' \sim \pi(a' \mid s')} \pi\left(a' \mid s'\right)\left(Q^{\pi}\left(s',a'\right) - \alpha \log \pi\left(a' \mid s'\right)\right)\right] \qquad \text{discrete action space} \tag{124}$$

Reference

1. PG connection with Q-learning

   Equivalence Between Policy Gradients and Soft Q-Learning

   https://arxiv.org/pdf/1704.06440.pdf

2. GAE

   HIGH-DIMENSIONAL CONTINUOUS CONTROL USING GENERALIZED ADVANTAGE ESTIMATION

   https://arxiv.org/pdf/1506.02438.pdf

3. Natural gradient

   Trust Region Policy Optimization

   https://arxiv.org/pdf/1502.05477v5.pdf

4. DQN with improvements

   Rainbow: Combining Improvements in Deep Reinforcement Learning

   https://arxiv.org/pdf/1710.02298.pdf

5. SAC discrete

   SOFT ACTOR-CRITIC FOR DISCRETE ACTION SETTINGS

   https://arxiv.org/pdf/1910.07207.pdf

6. approximation of pg theorem for solving off-policy case

   Off-Policy Actor-Critic

   https://arxiv.org/pdf/1205.4839.pdf

7. a great summary for many pg algorithm

   Policy Gradient Algorithms

   https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html#off-policy-policy-gradient

8. detail post about basic RL concept A (Long) Peek into Reinforcement Learning

   https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html

9. regularized policy gradient techniques can be interpreted as advantage function learning algorithms

   COMBINING POLICY GRADIENT AND Q-LEARNING

   https://openreview.net/pdf?id=B1kJ6H9ex

10. greate introduction of RL algorithm and background

    Deep Reinforcement Learning

    https://julien-vitay.net/deeprl/Introduction.html#sec:introduction

11. a vivid blog post of natural gradient

    RL — Natural Policy Gradient Explained

    https://jonathan-hui.medium.com/rl-natural-policy-gradient-actor-critic-using-kronecker-factored-trust-region-acktr-58f3798a4a93

12. RL framework which is easy to use and have fantastic documentation

    https://spinningup.openai.com/en/latest/index.html

13. CS294-112a, basically everything you need to know about Rl

    http://rail.eecs.berkeley.edu/deeprlcourse-fa17/

14. if only one course I could recommend, this is the one

    UCL Course on RL

    https://www.davidsilver.uk/teaching/

15. beat human on all atari games

    Agent57: Outperforming the Atari Human Benchmark

    https://arxiv.org/pdf/2003.13350.pdf

16. a new approach dealing with exploration problem Rethinking Exploration for Sample-Efficient Policy Learning https://arxiv.org/pdf/2101.09458.pdf