Maximum Entropy Reinforcement Learning


Reinforcement Learning

Deterministic Policies
- There always exists an optimal deterministic policy
- Search space is smaller for deterministic than stochastic policies
- Practitioners prefer deterministic policies

Stochastic Policies
- Search space is continuous for stochastic policies (helps with gradient descent)
- More robust (less likely to overfit)
- Naturally incorporate exploration
- Facilitate transfer learning
- Mitigate local optima
Encouraging Stochasticity

**Standard MDP**
- States: $S$
- Actions: $A$
- Reward: $R(s, a)$
- Transition: $\Pr(s' | s, a)$
- Discount: $\gamma$

**Soft MDP**
- States: $S$
- Actions: $A$
- Reward: $R(s, a) + \lambda H(\pi(\cdot | s))$
- Transition: $\Pr(s' | s, a)$
- Discount: $\gamma$
Optimal Policy

- Standard MDP
  \[ \pi^* = \arg\max_{\pi} \sum_{n=0}^{N} \gamma^n E_{s_n,a_n|\pi}[R(s_n,a_n)] \]

- Soft MDP
  \[ \pi^*_{soft} = \arg\max_{\pi} \sum_{n=0}^{N} \gamma^n E_{s_n,a_n|\pi}[R(s_n,a_n) + \lambda H(\pi(\cdot | s_n))] \]

Maximum entropy policy
Entropy regularized policy
Q-function

• Standard MDP

\[
Q^\pi(s_0, a_0) = R(s_0, a_0) + \sum_{n=1}^{\infty} \gamma^n E_{s_n, a_n|s_0, a_0, \pi}[R(s_n, a_n)]
\]

• Soft MDP

\[
Q^\pi_{soft}(s_0, a_0) = R(s_0, a_0) + \sum_{n=1}^{\infty} \gamma^n E_{s_n, a_n|s_0, a_0, \pi}[R(s_n, a_n) + \lambda H(\pi(\cdot | s_n))]
\]

NB: No entropy with first reward term since action is not chosen according to $\pi$
Greedy Policy

- Standard MDP (deterministic policy)
  \[ \pi_{\text{greedy}}(s) = \arg\max_a Q(s, a) \]

- Soft MDP (stochastic policy)
  \[ \pi_{\text{greedy}}(\cdot \mid s) = \arg\max_{\pi} \sum_a \pi(a \mid s)Q(s, a) + \lambda H(\pi(\cdot \mid s)) \]
  \[ = \frac{\exp(Q(s, \cdot)/\lambda)}{\sum_a \exp(Q(s,a)/\lambda)} = \text{softmax}(Q(s, \cdot)/\lambda) \]

  when \( \lambda \to 0 \) then \( \text{softmax} \) becomes regular max
Soft Policy Iteration

\( \text{SoftPolicyIteration}(\text{MDP}, \lambda) \)

Initialize \( \pi_0 \) to any policy

\( i \leftarrow 0 \)

Repeat

Policy evaluation:
Repeat until convergence

\[
Q_{\text{soft}}^{\pi_i}(s, a) \leftarrow R(s, a) + \gamma \sum_{s'} \Pr(s'|s, a) \left[ \sum_{a'} \pi_i(a'|s')Q_{\text{soft}}^{\pi_i}(s', a') + \lambda H(\pi_i(\cdot|s')) \right] \quad \forall s, a
\]

Policy improvement:

\[
\pi_{i+1}(a|s) \leftarrow \text{softmax} \left( Q_{\text{soft}}^{\pi_i}(s, a)/\lambda \right) = \frac{\exp(Q_{\text{soft}}^{\pi_i}(s, a)/\lambda)}{\sum_{a'} \exp(Q_{\text{soft}}^{\pi_i}(s, a')/\lambda)} \quad \forall s, a
\]

\( i \leftarrow i + 1 \)

Until \( \left\| Q_{\text{soft}}^{\pi_i}(s, a) - Q_{\text{soft}}^{\pi_{i-1}}(s, a) \right\|_\infty \leq \epsilon \)
Soft Actor-Critic

- RL version of soft policy iteration
- Use neural networks to represent policy and value function
- At each policy improvement step, project new policy in the space of parameterized neural nets
Soft Actor Critic (SAC)

Initialize weights \( w, \bar{w}, \theta \) at random in \([-1,1]\)
Observe current state \( s \)
Loop

\[ \begin{align*}
\text{Sample action } a &\sim \pi_\theta (\cdot \mid s) \text{ and execute it} \\
\text{Receive immediate reward } r \\
\text{Observe new state } s' \\
\text{Add } (s, a, s', r) \text{ to experience buffer} \\
\text{Sample mini-batch of experiences from buffer} \\
\text{For each experience } (\hat{s}, \hat{a}, \hat{s}', \hat{r}) \text{ in mini-batch} \\
\quad \text{Sample } \hat{a}' \sim \pi_\theta (\cdot \mid \hat{s}') \\
\text{Gradient: } \frac{\partial \text{Err}}{\partial w} = \left[ Q_w^{soft} (\hat{s}, \hat{a}) - \hat{r} - \gamma [Q_w^{soft} (\hat{s}', \hat{a}') + \lambda H (\pi_\theta (\cdot \mid \hat{s}'))] \right] \frac{\partial Q_w^{soft} (\hat{s}, \hat{a})}{\partial w} \\
\text{Update weights: } w \leftarrow w - \alpha \frac{\partial \text{Err}}{\partial w} \\
\text{Update policy: } \theta \leftarrow \theta - \alpha \frac{\partial \text{KL}(\pi_\theta \mid \text{softmax}(Q_w^{soft} / \lambda))}{\partial \theta} \\
\text{Update state: } s \leftarrow s' \\
\text{Every } c \text{ steps, update target: } \bar{w} \leftarrow w
\end{align*} \]