Lecture 6b: Maximum Entropy RL
CS885 Reinforcement Learning

2022-09-30

Complementary readings:
Haarnoja, Tang, Abbeel, Levine (2017) Reinforcement Learning with Deep Energy-Based Policies, ICML.
Haarnoja, Zhou, Abbeel, Levine (2018) Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, ICML.

Pascal Poupart
David R. Cheriton School of Computer Science

UNIVERSITY OF WATERLOO

Maximum Entropy RL

- Why do several implementations of important RL baselines (e.g., A2C, PPO) add an entropy regularizer?
- Why is maximizing entropy desirable in RL?
- What is the Soft Actor Critic algorithm?

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: γ

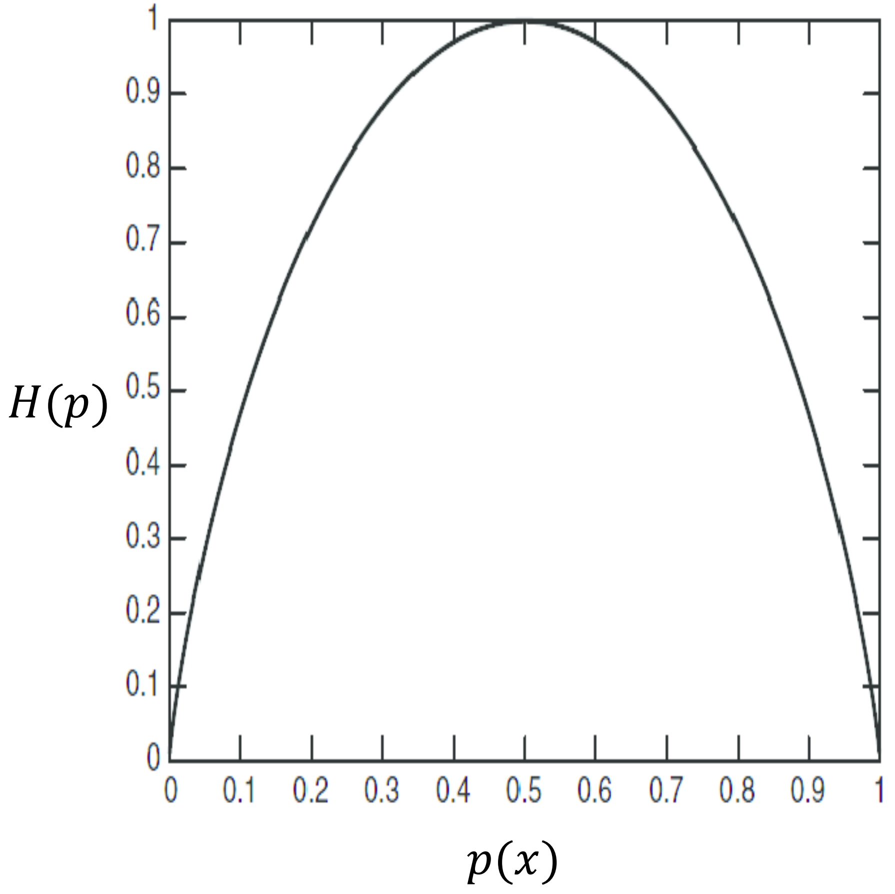
Soft MDP

- States: S
- Actions: A
- Reward: $R(s,a) + \lambda H\big(\pi(\cdot|s)\big)$
- Transition: $\Pr(s'|s,a)$
- Discount: γ

Entropy

- Measure of uncertainty
  - Information theory: expected # of bits needed to communicate the result of a sample

$$H(p) = -\sum_x p(x) \log p(x)$$



Optimal Policy

- Standard MDP: $\pi^* = \underset{\pi}{\text{argmax}} \sum_{n=0}^{N} \gamma^n E_{s_n, a_n | \pi}[R(s_n, a_n)]$

- Soft MDP: $\pi^*_{soft} = \underset{\pi}{\text{argmax}} \sum_{n=0}^{N} \gamma^n E_{s_n, a_n | \pi}\left[R(s_n, a_n) + \lambda H\left(\pi(\cdot | s_n)\right)\right]$

Maximum entropy policy
Entropy regularized policy

UNIVERSITY OF WATERLOO

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

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

NB: No entropy with first reward term
since action is not chosen according to π

Greedy Policy

- Standard MDP (deterministic policy)

$$\pi_{greedy}(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

- Soft MDP (stochastic policy)

$$\pi_{greedy}(\cdot|s) = \underset{\pi}{\operatorname{argmax}} \sum_a \pi(a|s)Q(s,a) + \lambda H\big(\pi(\cdot|s)\big)$$

$$= \frac{\exp(Q(s,\cdot)/\lambda)}{\sum_a \exp(Q(s,a)/\lambda)} = softmax(Q(s,\cdot)/\lambda)$$

when $\lambda \to 0$ then $softmax$ becomes regular max

Derivation

- Concave objective (can find global maximum)

$$J(\pi, Q) = \sum_a \pi(a|s) Q(s,a) + \lambda H\big(\pi(\cdot\,|s)\big)$$

$$= \sum_a \pi(a|s)[Q(s,a) - \lambda \log \pi(a|s)]$$

- Partial derivative: $\frac{\partial J}{\partial \pi(a|s)} = Q(s,a) - \lambda[\log \pi(a|s) + 1]$

- Setting the derivative to 0 and isolating $\pi(a|s)$ yields

$$\pi(a|s) = \exp(Q(s,a)/\lambda - 1) \propto \exp(Q(s,a)/\lambda)$$

- Hence $\pi_{greedy}(\cdot\,|s) = \frac{\exp(Q(s,\cdot)/\lambda)}{\sum_a \exp(Q(s,a)/\lambda)} = softmax(Q(s,\cdot)/\lambda)$

Greedy Value Function

- What is the value function induced by the greedy policy?
- Standard MDP: $V(s) = \max_a Q(s, a)$
- Soft MDP: $V_{soft}(s) = \lambda H\left(\pi_{greedy}(\cdot\,|s)\right) + \sum_a \pi_{greedy}(a|s) Q_{soft}(s, a)$

$$= \lambda \log \sum_a \exp\left(\frac{Q_{soft}(s,a)}{\lambda}\right) = \widetilde{max}_{\lambda,a}\, Q_{soft}(s, a)$$

when $\lambda \to 0$ then $\widetilde{max}_\lambda$ becomes regular max

Derivation

$$V_{soft}(s) = \lambda H\left(\pi_{greedy}(\cdot|s)\right) + \sum_a \pi_{greedy}(a|s) Q_{soft}(s,a)$$

since $\pi_{greedy}(a|s) = \frac{\exp\left(Q_{soft}(s,a)/\lambda\right)}{\sum_{a'} \exp\left(Q_{soft}(s,a')/\lambda\right)}$

$$= \lambda H\left(\pi_{greedy}(\cdot|s)\right) + \sum_a \pi_{greedy}(a|s)\lambda\left[\log \pi_{greedy}(a|s) + \log \sum_{a'} \exp\left(\frac{Q_{soft}(s,a')}{\lambda}\right)\right]$$

$$= \lambda H\left(\pi_{greedy}(\cdot|s)\right) + \lambda \sum_a \pi_{greedy}(a|s) \log \pi_{greedy}(a|s) \; + \lambda \log \sum_{a'} \exp\left(\frac{Q_{soft}(s,a')}{\lambda}\right)$$

$$= \lambda H\left(\pi_{greedy}(\cdot|s)\right) - \lambda H\left(\pi_{greedy}(\cdot|s)\right) + \lambda \log \sum_{a'} \exp\left(\frac{Q_{soft}(s,a')}{\lambda}\right)$$

$$= \lambda \log \sum_{a'} \exp\left(\frac{Q_{soft}(s,a')}{\lambda}\right)$$

$$= \underset{a}{\widetilde{\max}}_{\lambda}\, Q_{soft}(s,a)$$

UNIVERSITY OF WATERLOO

Soft Q-Value Iteration

SoftQValueIteration(MDP, λ)

Initialize π_0 to any policy

$i \leftarrow 0$

Repeat

$$Q^{i+1}_{soft}(s,a) \leftarrow R(s,a) + \gamma \sum_{s'} \Pr(s'|s,a) \underset{a'}{\widetilde{\max}}_{\lambda} Q^{i}_{soft}(s',a')$$

$i \leftarrow i + 1$

Until $\left|\left|Q^{i}_{soft}(s,a) - Q^{i-1}_{soft}(s,a)\right|\right|_{\infty} \leq \epsilon$

Extract policy: $\pi_{greedy}(\cdot|s) = softmax(Q^{i}_{soft}(s,\cdot)/\lambda)$

Soft Bellman equation: $Q^{*}_{soft}(s,a) = R(s,a) + \gamma \sum_{s'} \Pr(s'|s,a) \underset{a'}{\widetilde{\max}}_{\lambda} Q^{*}_{soft}(s',a')$

Soft Q-learning

- Q-learning based on Soft Q-Value Iteration
- Replace expectations by samples
- Represent Q-function by a function approximator (e.g., neural network)
- Do gradient updates based on temporal differences

Soft Q-learning (Soft Variant of DQN)

Initialize weights $\boldsymbol{w}$ and $\overline{\boldsymbol{w}}$ at random in $[-1,1]$
Observe current state s
Loop
    Select action a and execute it
    Receive immediate reward r, observe new state s'
    Add (s, a, s', r) to experience buffer
    Sample mini-batch of experiences from buffer
    For each experience $(\hat{s}, \hat{a}, \hat{s}', \hat{r})$ in mini-batch
        Gradient: $\frac{\partial Err}{\partial \boldsymbol{w}} = \left[Q_{\boldsymbol{w}}^{soft}(\hat{s}, \hat{a}) - \hat{r} - \gamma \, \widetilde{\max}_{\lambda,\hat{a}'} \, Q_{\overline{\boldsymbol{w}}}^{soft}(\hat{s}', \hat{a}') \right] \frac{\partial Q_{\boldsymbol{w}}^{soft}(\hat{s},\hat{a})}{\partial \boldsymbol{w}}$
        Update weights: $\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \frac{\partial Err}{\partial \boldsymbol{w}}$
    Update state: $s \leftarrow s'$
    Every c steps, update target: $\overline{\boldsymbol{w}} \leftarrow \boldsymbol{w}$

Soft Actor Critic

- In practice, actor critic techniques tend to perform better than Q-learning.
- Can we derive a soft actor-critic algorithm?

- Yes, idea:
  - Critic: soft Q-function
  - Actor: (greedy) softmax policy

UNIVERSITY OF WATERLOO

Soft Policy Iteration

Initialize π_0 to any policy, $i \leftarrow 0$

Repeat

Policy evaluation:

Repeat until convergence $\quad \forall s, a$

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

Policy improvement:

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

$i \leftarrow i + 1$

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

Policy Improvement

Theorem 1: Let $Q^{\pi_i}_{soft}(s,a)$ be the Q-function of π_i
Let $\pi_{i+1}(a|s) = softmax\left(Q^{\pi_i}_{soft}(s,a)/\lambda\right)$
Then $Q^{\pi_{i+1}}_{soft}(s,a) \geq Q^{\pi_i}_{soft}(s,a)\ \forall s,a$

Proof: first show that

$$\sum_a \pi_i(a|s) Q^{\pi_i}_{soft}(s,a) + \lambda H\big(\pi_i(\cdot\,|s)\big) \leq \sum_a \pi_{i+1}(a|s) Q^{\pi_i}_{soft}(s,a) + \lambda H(\pi_{i+1}(\cdot\,|s)$$

then use this inequality to show that

$$Q^{\pi_{i+1}}_{soft}(s,a) \geq Q^{\pi_i}_{soft}(s,a)\ \forall s,a$$

Inequality Derivation

$$\sum_a \pi_i(a|s) Q^{\pi_i}_{soft}(s,a) + \lambda H\big(\pi_i(\cdot\,|s)\big)$$

$$= \sum_a \pi_i(a|s) \left[Q^{\pi_i}_{soft}(s,a) - \lambda \log \pi_i(a|s)\right] \quad \text{since } \pi_{i+1}(a|s) = \frac{\exp\left(Q^{\pi_i}_{soft}(s,a)/\lambda\right)}{\sum_{a'} \exp\left(Q^{\pi_i}_{soft}(s,a')/\lambda\right)}$$

$$= \sum_a \pi_i(a|s)[\lambda \log \pi_{i+1}(a|s) - \lambda \log \textstyle\sum_{a'} \exp(Q^{\pi_i}_{soft}(s,a')/\lambda) - \lambda \log \pi_i(a|s)]$$

$$= \lambda \sum_a \pi_i(a|s)[\log \frac{\pi_{i+1}(a|s)}{\pi_i(a|s)} + \log \textstyle\sum_{a'} \exp(Q^{\pi_i}_{soft}(s,a')/\lambda)]$$

$$= -\lambda KL(\pi_{i+1}||\pi_i) + \lambda \sum_a \pi_i(a|s) \log \textstyle\sum_{a'} \exp(Q^{\pi_i}_{soft}(s,a')/\lambda)$$

$$\leq \lambda \sum_a \pi_i(a|s) \log \textstyle\sum_{a'} \exp(Q^{\pi_i}_{soft}(s,a')/\lambda)$$

$$= \sum_a \pi_{i+1}(a|s)\, \lambda \log \textstyle\sum_{a'} \exp(Q^{\pi_i}_{soft}(s,a')/\lambda) \quad \text{since } \pi_{i+1}(a|s) = \frac{\exp(Q^{\pi_i}(s,a)/\lambda)}{\sum_{a'} \exp(Q^{\pi_i}(s,a')/\lambda)}$$

$$= \sum_a \pi_{i+1}(a|s) \left[Q^{\pi_i}_{soft}(s,a) - \lambda \log \pi_{i+1}(s,a)\right]$$

$$= \sum_a \pi_{i+1}(a|s)\, Q^{\pi_i}_{soft}(s,a) + \lambda H\big(\pi_{i+1}(\cdot\,|s)\big)$$

UNIVERSITY OF WATERLOO

Proof Derivation

$$Q^{\pi_i}_{soft}(s,a) = R(s,a) + \gamma E_{s'}\left[E_{a'\sim\pi_i}\left[Q^{\pi_i}_{soft}(s',a')\right] + \lambda H\big(\pi_i(\cdot\,|s')\big)\right]$$

since $E_{a'\sim\pi_i}\left[Q^{\pi_i}_{soft}(s',a')\right] + \lambda H\big(\pi_i(\cdot\,|s')\big) \leq E_{a'\sim\pi_{i+1}}\left[Q^{\pi_i}_{soft}(s',a')\right] + \lambda H\big(\pi_{i+1}(\cdot\,|s')\big)$

$$\leq R(s,a) + \gamma E_{s'}\left[E_{a'\sim\pi_{i+1}}\left[Q^{\pi_i}_{soft}(s',a')\right] + \lambda H\big(\pi_{i+1}(\cdot\,|s')\big)\right]$$

$\leq \cdots$ repeatedly apply

$\leq \cdots$ $Q^{\pi_i}_{soft}(s',a') \leq R(s',a') + \gamma E_{s''}\left[E_{a''\sim\pi_{i+1}}\left[Q^{\pi_i}_{soft}(s'',a'')\right] + \lambda H\big(\pi_{i+1}(\cdot\,|s'')\big)\right]$

$$\leq Q^{\pi_{i+1}}_{soft}(s,a)$$

UNIVERSITY OF WATERLOO

Convergence to Optimal Q^*_{soft} and π^*_{soft}

- Theorem 2: When $\epsilon = 0$,
  soft policy iteration converges to optimal Q^*_{soft} and π^*_{soft}.

- Proof:
  - We know that $Q^{\pi_{i+1}}(s,a) \geq Q^{\pi_i}(s,a)\ \forall s,a$ according to Theorem 1
  - Since the Q-functions are upper bounded by $\left(\max_{s,a} R(s,a) + H(uniform)\right)/(1-\gamma)$
    then soft policy iteration converges
  - At convergence, $Q^{\pi_{i-1}} = Q^{\pi_i}$ and therefore the Q-function satisfies Bellman's equation:

$$Q^{\pi_{i-1}}_{soft}(s,a) = Q^{\pi_i}_{soft}(s,a) = R(s,a) + \gamma \sum_{s'} \Pr(s'|s,a)\, \widetilde{\max_{a'}}_{\lambda}\, Q^{\pi_{i-1}}_{soft}(s',a')$$

UNIVERSITY OF WATERLOO

Soft Actor-Critic

- RL version of soft policy iteration
- Use neural networks to represent policy and value functions
- At each policy improvement step, project new policy in the space of parameterized neural nets

Soft Actor-Critic (SAC)

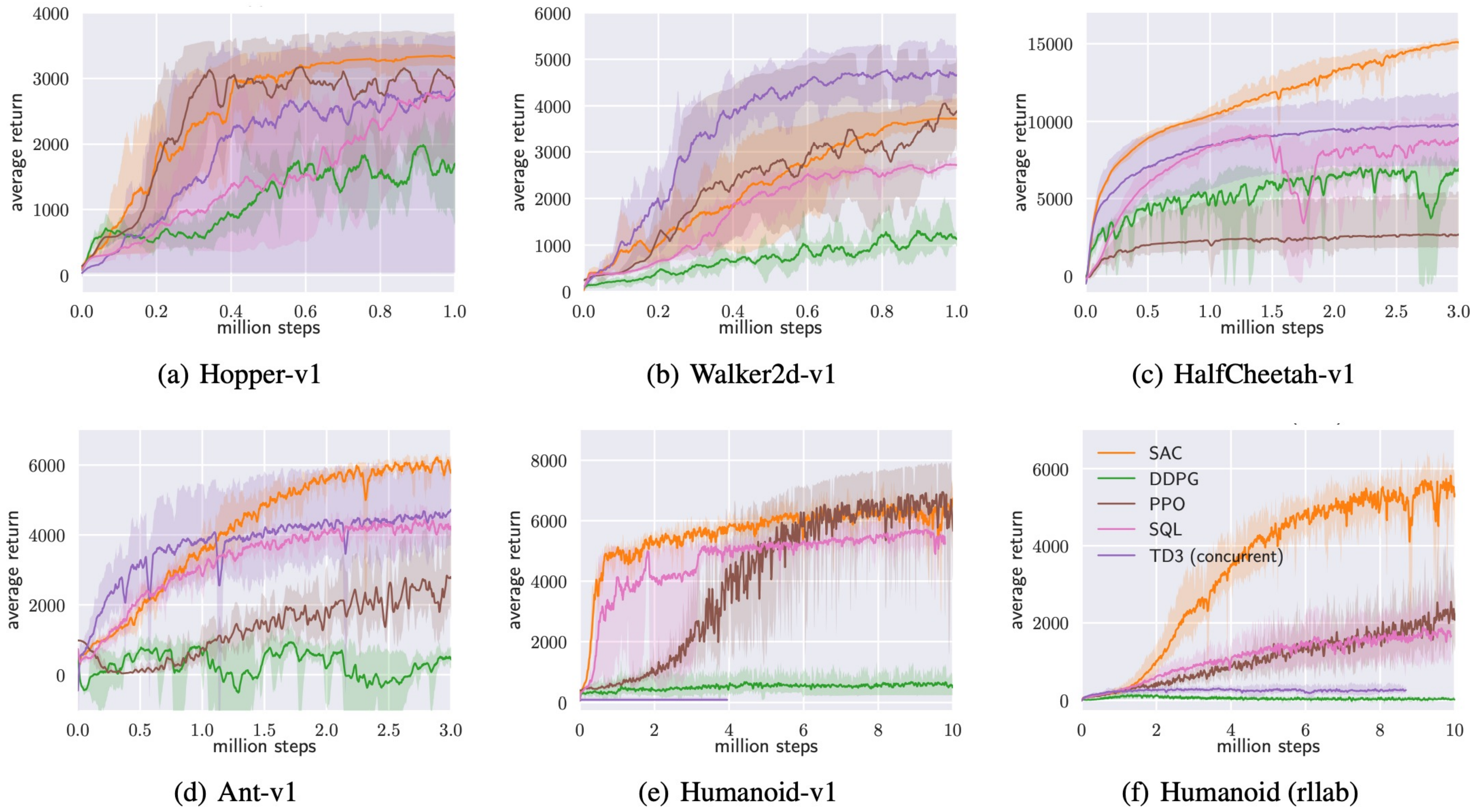
Initialize weights $\boldsymbol{w}$, $\overline{\boldsymbol{w}}$, θ at random in $[-1,1]$
Observe current state s
Loop
 Sample action $a \sim \pi_\theta(\cdot\,|s)$ and execute it
 Receive immediate reward r, observe new state s'
 Add (s, a, s', r) to experience buffer
 Sample mini-batch of experiences from buffer
 For each experience $(\hat{s}, \hat{a}, \hat{s}', \hat{r})$ in mini-batch
  Sample $\hat{a}' \sim \pi_\theta(\cdot\,|\hat{s}')$
  Gradient: $\frac{\partial Err}{\partial \boldsymbol{w}} = \left[Q_{\boldsymbol{w}}^{soft}(\hat{s}, \hat{a}) - \hat{r} - \gamma [Q_{\overline{\boldsymbol{w}}}^{soft}(\hat{s}', \hat{a}') + \lambda H(\pi_\theta(\cdot\,|\hat{s}'))] \right] \frac{\partial Q_{\boldsymbol{w}}^{soft}(\hat{s},\hat{a})}{\partial \boldsymbol{w}}$
  Update weights: $\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \frac{\partial Err}{\partial \boldsymbol{w}}$
  Update policy: $\theta \leftarrow \theta - \alpha \frac{\partial KL\left(\pi_\theta \middle| softmax\left(Q_{\overline{w}}^{soft}/\lambda\right)\right)}{\partial \theta}$
 Update state: $s \leftarrow s'$
 Every c steps, update target: $\overline{\boldsymbol{w}} \leftarrow \boldsymbol{w}$

UNIVERSITY OF WATERLOO

Empirical Results

From Haarnoja, Zhou et al. (2018)

Comparison on several robotics tasks



UNIVERSITY OF WATERLOO

Robustness to Environment Changes

Using Soft Actor Critic (SAC), Minotaur learns to walk quickly and to generalize to environments with challenges that it was not trained to deal with!

SAC on Minotaur - Testing