Carnegie Mellon
School of Computer Science

Deep Reinforcement Learning and Control

Maximum Entropy Reinforcement Learning

CMU 10-703

Katerina Fragkiadaki

RL objective

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi} \left[\sum_t R(s_t, a_t) \right]$$

The optimal policy is a deterministic policy.

Stochastic policies are useful

- Learn all possible ways of accomplishing a task
- Better policies to finetune from for specific environments
- Better exploration
- Composability of policies by adding their Q values
- Robustness to hyperparameters

We need an objective that promotes stochasticity..

MaxEntRL objective

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=1}^{T} \underbrace{R(s_t, a_t)}_{\text{reward}} + \alpha \underbrace{\mathrm{H}(\pi(\cdot \mid s_t))}_{\text{entropy}} \right]$$

- Learn all possible ways of accomplishing a task
- Better policies to finetune from for specific environments
- Better exploration
- Composability of policies by adding their Q values
- Robustness to hyperparameters

Principle of Maximum Entropy

Policies that generate similar rewards, should be equally probable.

We do not want to commit.

Reinforcement Learning with Deep Energy-Based Policies,Haarnoja et al.

Have we seen this before?

Algorithm S3 Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

// Assume global shared parameter vectors θ *and* θ_v *and global shared counter* $T = 0$
// Assume thread-specific parameter vectors θ' *and* θ'_v
Initialize thread step counter $t \leftarrow 1$
repeat
  Reset gradients: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.
  Synchronize thread-specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$
  $t_{start} = t$
  Get state s_t
  repeat
    Perform a_t according to policy $\pi(a_t|s_t;\theta')$
    Receive reward r_t and new state s_{t+1}
    $t \leftarrow t + 1$
    $T \leftarrow T + 1$
  until terminal s_t **or** $t - t_{start} == t_{max}$
  $R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{for non-terminal } s_t \text{// Bootstrap from last state} \end{cases}$
  for $i \in \{t-1, \ldots, t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$
    Accumulate gradients wrt θ': $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i \,|\, s_i;\theta')\big(R - V(s_i;\theta'_v)\big) + \alpha \nabla_{\theta'} H\left(\pi(s_i;\theta')\right)$
    Accumulate gradients wrt θ'_v: $d\theta_v \leftarrow d\theta_v + \partial \left(R - V(s_i;\theta'_v)\right)^2 / \partial \theta'_v$
  end for
  Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
until $T > T_{max}$

$$\mathrm{H}\left(\pi(\,\cdot\,|\,s)\right) = -\sum_a \pi(a\,|\,s)\log \pi(a\,|\,s)$$

"We also found that adding the entropy of the policy π to the objective function improved exploration by discouraging premature convergence to suboptimal deterministic policies. This technique was originally proposed by (Williams & Peng, 1991)"

Mnih et al., Asynchronous Methods for Deep Reinforcement Learning

Entropy Regularized policy gradient

$$\Delta\theta = \mathbb{E}_{s,a} A^{\pi}(s,a) \nabla_{\theta} \log \pi(a \mid s) + \alpha \mathbb{E}_{s} \nabla_{\theta} \mathrm{H}(\pi(s))$$

Prevents the policy from becoming deterministic, improves exploration Mnih et al. 2016, Williams and Peng 1991.

Entropy Regularized policy gradient

$$\Delta\theta = \mathbb{E}_{s,a} A^{\pi}(s,a) \nabla_{\theta} \log \pi(a \mid s) + \alpha \mathbb{E}_{s} \nabla_{\theta} \mathrm{H}(\pi(s))$$

Prevents the policy from becoming deterministic, improved exploration Mnih et al. 2016, Williams and Peng 1991,

Adding an entropy gradient term maximizes the entropy of the policy *in the current time step*. The maximum entropy RL objective seeks policies that lead to (future) states where the policy would have high entropy.

Boltzmann exploration

Instead of using argmax, use softmax over estimated Q values (of the standard RL objective), then sample actions based on the resulting action probability distribution:

$$P_t(a_i \mid s) = \frac{\exp\left(Q_t(s, a_i \mid \theta)/\tau\right)}{\sum_{j=1}^{n} \exp\left(Q_t(s, a_j \mid \theta)/\tau\right)}$$

as $\tau \to 0$ policy becomes deterministic

Discrete actions.

Actions that have same Q values have equal probabilities to be selected.

This greedily maximizes the entropy of the *current* time step, but not future timesteps.

Energy-based policies

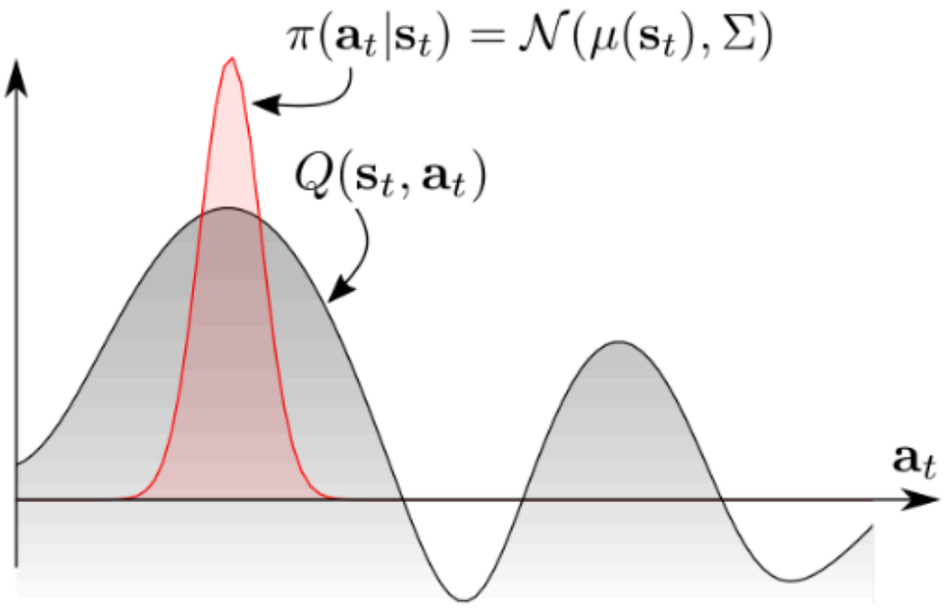
Instead of using argmax, use softmax over estimated Q values (of the entropy augmeted objective), then sample actions based on the resulting action probability distribution:

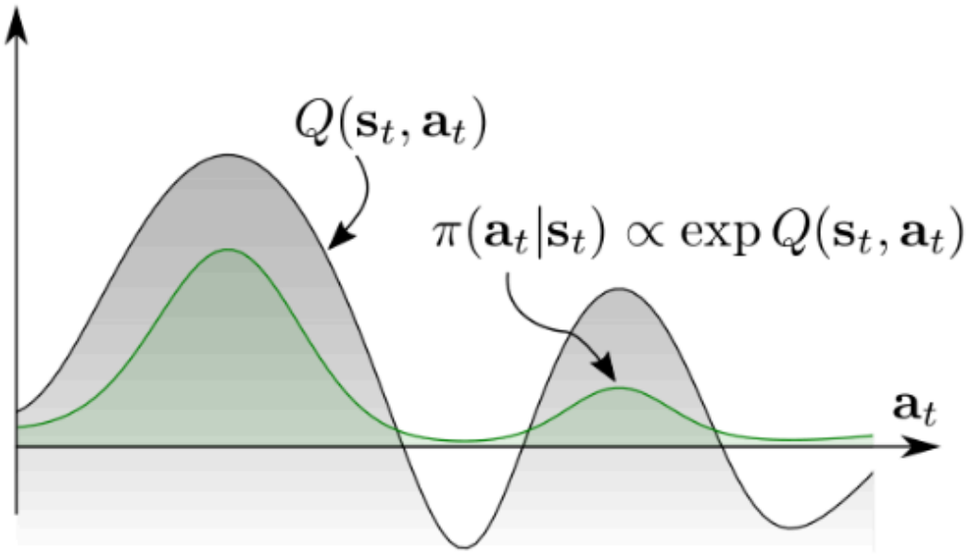
$$\pi(a_t \mid s_t) \propto \exp(\frac{1}{a} Q^{soft}(s_t, a_t))$$

Q_π^{soft}: expected cumulative (discounted) sum of rewards and expected cumulative (discounted) sum of entropies of the state conditioned action distributions.

Energy-based policies

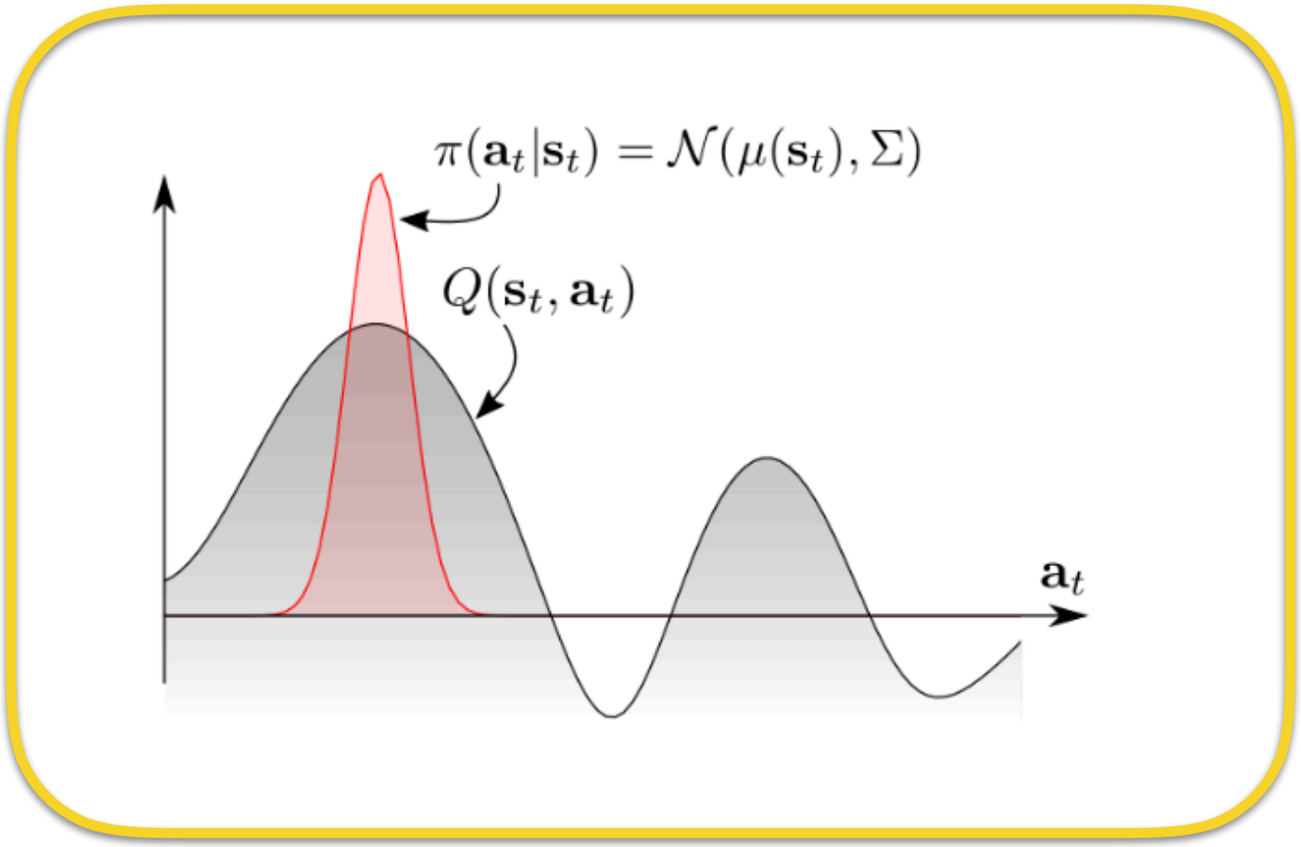
We need an expressive parametrization for our policy to be able to capture the different modes of our soft Q function.

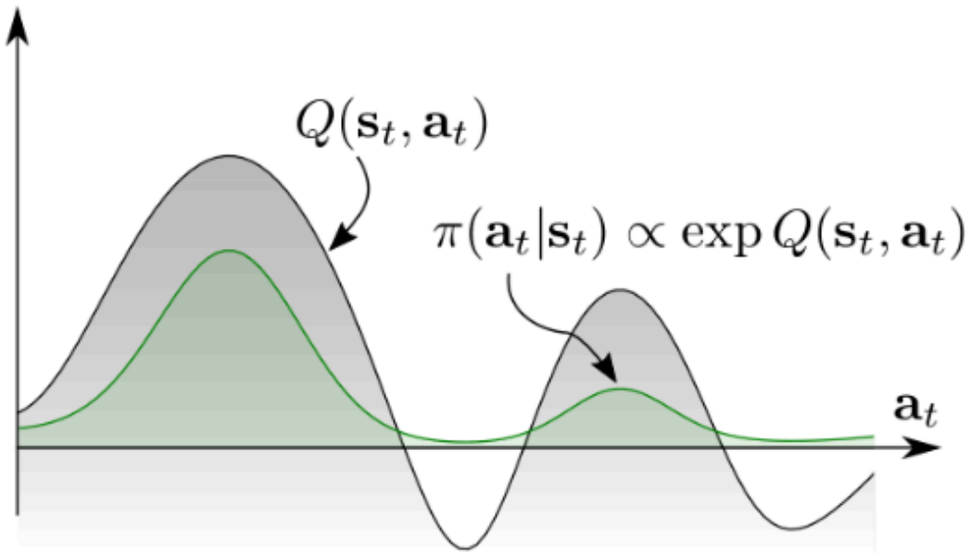




Energy-based policies

We need an expressive parametrization for our policy to be able to capture the different modes of our soft Q function.





For now we will limit ourselves to Gaussian policies..

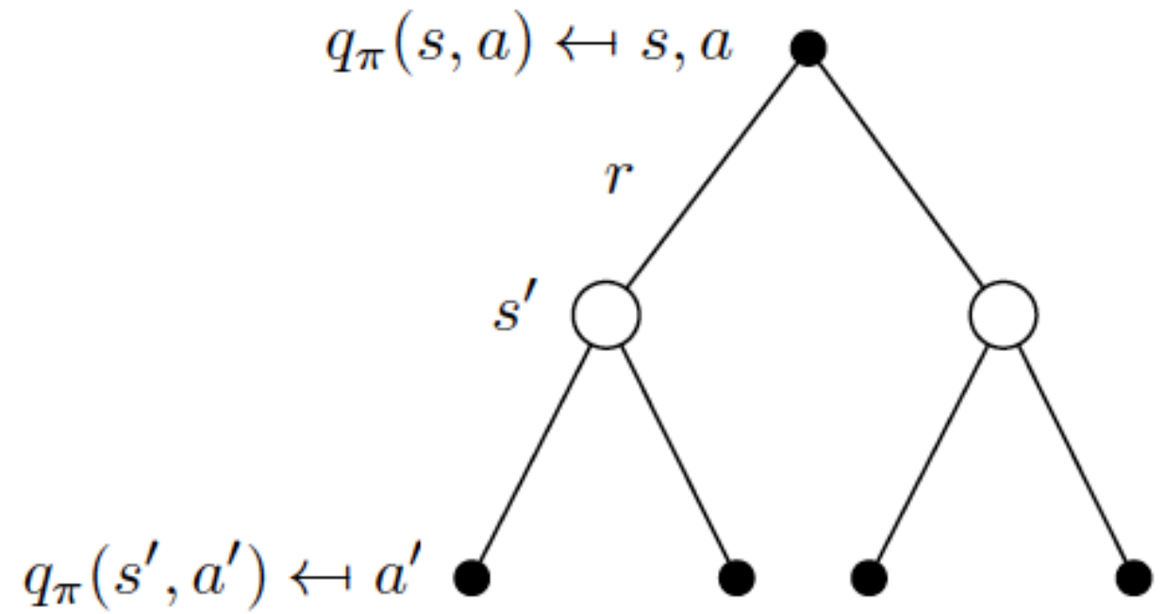
MaxEntRL objective

Promoting stochastic policies

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=1}^{T} \underbrace{R(s_t, a_t)}_{\text{reward}} + \alpha \underbrace{\mathrm{H}(\pi(\cdot \mid s_t))}_{\text{entropy}} \right]$$

Q_{π}^{soft}: expected cumulative (discounted) sum of rewards and expected cumulative (discounted) sum of entropies of the state conditioned action distributions.

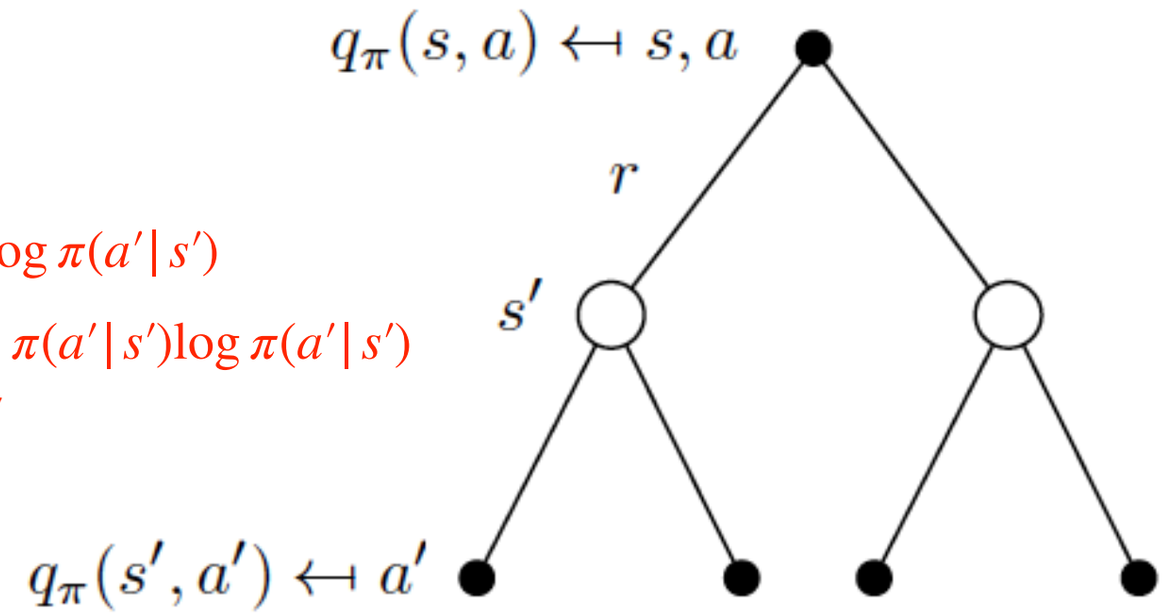
Recall:Back-up Diagrams



$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \sum_{a' \in \mathcal{A}} \pi(a' \mid s') q_\pi(s',a')$$

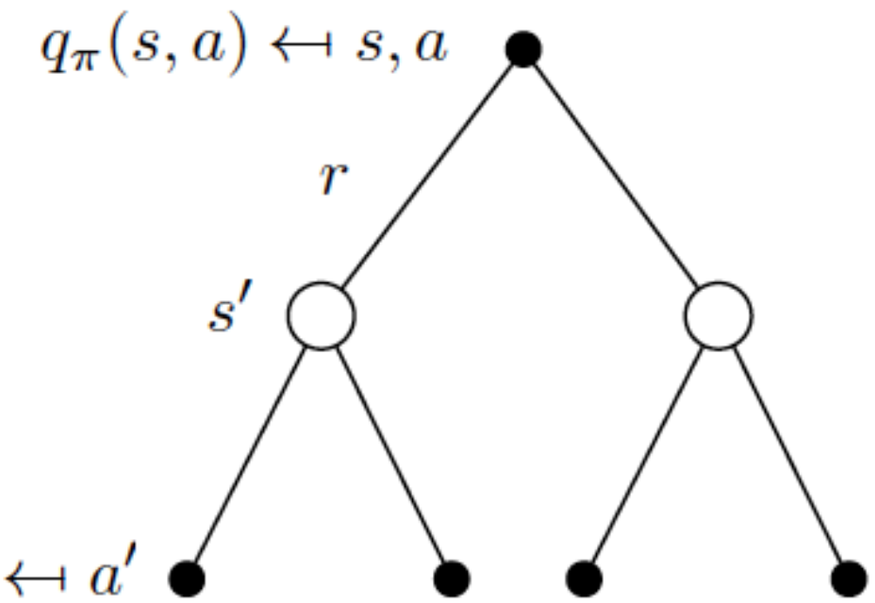
Back-up Diagrams for MaxEnt Objective

$$\begin{aligned}\alpha H(\pi(\cdot\,|\,s')) &= -\alpha \mathbb{E}_{a'} \log \pi(a'|s') \\ &= -\alpha \sum_{a' \in \mathcal{A}} \pi(a'|s') \log \pi(a'|s')\end{aligned}$$



$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s,a) \left(\alpha H(\pi(\cdot\,|\,s')) + \sum_{a' \in \mathcal{A}} \pi(a'|s') q_\pi(s',a') \right)$$

Back-up Diagrams for MaxEnt Objective

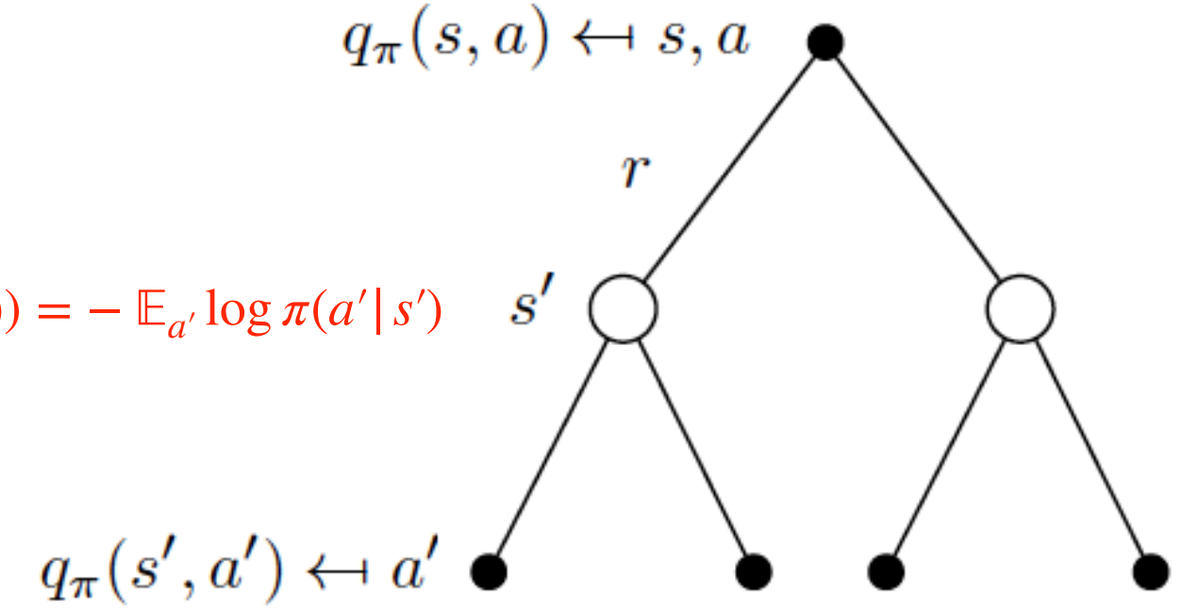


$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \left(\alpha H(\pi(\cdot \mid s')) + \sum_{a' \in \mathcal{A}} \pi(a' \mid s') q_\pi(s',a') \right)$$

$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \sum_{a' \in \mathcal{A}} \pi(a' \mid s') \left(q_\pi(s',a') - \alpha \log(\pi(a' \mid s')) \right)$$

Back-up Diagrams for MaxEnt Objective

$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \sum_{a' \in \mathcal{A}} \pi(a' \mid s') \big(q_\pi(s',a') - \alpha \log(\pi(a' \mid s')) \big)$$



$$H(\pi(\cdot \mid s')) = -\,\mathbb{E}_{a'} \log \pi(a' \mid s')$$

$$q^\pi(s_t,a_t) = r(s_t,a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho(\cdot \mid s_t, a_t)}[V(s_{t+1})]$$

$$V(s_t) = \mathbb{E}_{a_t \sim \pi(\cdot \mid s_t)} q^\pi(s_t,a_t) + \alpha H(\pi(\cdot \mid s_t)) = \mathbb{E}_{a_t \sim \pi(\cdot \mid s_t)} \left[q^\pi(s_t,a_t) - \alpha \log \pi(a_t \mid s_t) \right]$$

(Soft) policy evaluation

Bellman equation:

$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \sum_{a' \in \mathcal{A}} \pi(a' \mid s') q_\pi(s',a')$$

Soft Bellman equation:

$$q_\pi^{soft}(s,a) = r(s,a) + \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \sum_{a' \in \mathcal{A}} \left(q_\pi^{soft}(s',a') - \alpha \log(\pi(a' \mid s')) \right)$$

Bellman backup update operator:

$$Q(s_t,a_t) \leftarrow r(s_t,a_t) + \gamma \mathbb{E}_{s_{t+1},a_{t+1}}[Q(s_{t+1},a_{t+1} \mid s_{t+1})]$$

Soft Bellman backup update operator:

$$Q^{soft}(s_t,a_t) \leftarrow r(s_t,a_t) + \gamma \mathbb{E}_{s_{t+1},a_{t+1}} \left[Q^{soft}(s_{t+1},a_{t+1}) - \alpha \log(\pi(a_{t+1} \mid s_{t+1})) \right]$$

Soft Bellman backup update operator is a contraction

Starting from any Q assignment, applying soft Bellman Backup Operator iteratively converges to the soft Q value of the policy π.

$$Q(s_t, a_t) \leftarrow r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}, a_{t+1}} \left[Q(s_{t+1}, a_{t+1}) - \alpha \log(\pi(a_{t+1} \mid s_{t+1})) \right]$$

$$\begin{aligned} Q(s_t, a_t) &\leftarrow r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho}[\mathbb{E}_{a_{t+1} \sim \pi}[Q(s_{t+1}, a_{t+1}) - \alpha \log(\pi(a_{t+1} \mid s_{t+1}))]] \\ &\leftarrow r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho, a_{t+1} \sim \pi} Q(s_{t+1}, a_{t+1}) + \gamma \mathbb{E}_{s_{t+1} \sim \rho} \mathbb{E}_{a_{t+1} \sim \pi}[-\alpha \log \pi(a_{t+1} \mid s_{t+1}) \\ &\leftarrow r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho, a_{t+1} \sim \pi} Q(s_{t+1}, a_{t+1}) + \gamma \alpha \mathbb{E}_{s_{t+1} \sim \rho} \mathrm{H}(\pi(\cdot \mid s_{t+1})) \\ &\leftarrow r_{soft}(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho, a_{t+1} \sim \pi} Q(s_{t+1}, a_{t+1}) \end{aligned}$$

We rewrote the total reward as: $r_{soft}(s_t, a_t) = r(s_t, a_t) + \gamma \alpha \mathbb{E}_{s_{t+1} \sim \rho} \mathrm{H}(\pi(\cdot \mid s_{t+1}))$

Then we get the old Bellman operator, which we know is a contraction.

Review: Policy Iteration for deterministic policies

Policy iteration iterates between two steps:

1. Policy evaluation: Fix policy, apply Bellman backup operator till convergence

$$q(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{s',a'} q(s',a')$$

This converges to q_π

Review: Policy Iteration for deterministic policies

Policy iteration iterates between two steps:

1. Policy evaluation: Fix policy, apply Bellman backup operator till convergence

$$q(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{s',a'} q(s',a')$$

This converges to q_π

2. Policy improvement: Update the policy

$$\pi'(s) \doteq \arg\max_a q_\pi(s,a)$$

Then: $$q_\pi(s,\pi'(s)) = \arg\max_a q_\pi(s,a) \geq \sum_a \pi(a\,|\,s) q_\pi(s,a) = v_\pi(s)\,.$$

Review: Policy Improvement theorem for deterministic policies

Let π, π' be any pair of determinstic policies such that, for all $s \in \mathcal{S}$:

$$q_\pi(s, \pi'(s)) \geq v_\pi(s) .$$

Then π' must be as good as or better than π, that is:

$$v_{\pi'(s)} \geq v_{\pi(s)}$$

Review: Policy Improvement theorem for deterministic policies

Let π, π' be any pair of determinstic policies such that, for all $s \in \mathcal{S}$:

$$q_\pi(s, \pi'(s)) \geq v_\pi(s) .$$

Then π' must be as good as or better than π, that is:

$$v_{\pi'(s)} \geq v_{\pi(s)}$$

$$\begin{aligned}
v_\pi(s) &\leq q_\pi(s, \pi'(s)) \\
&= \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = \pi'(s)] && \text{(by (4.6))} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \mid S_t = s] && \text{(by (4.7))} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}_{\pi'}[R_{t+2} + \gamma v_\pi(S_{t+2}) | S_{t+1}, A_{t+1} = \pi'(S_{t+1})] \mid S_t = s] \\
&= \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) \;\middle|\; S_t = s\right] \\
&\leq \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(S_{t+3}) \;\middle|\; S_t = s\right] \\
&\vdots \\
&\leq \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \;\middle|\; S_t = s\right] \\
&= v_{\pi'}(s).
\end{aligned}$$

Soft Policy Iteration

Soft policy iteration iterates between two steps:

1. Soft policy evaluation: Fix policy, apply soft Bellman backup operator till convergence

$$Q(s,a) \leftarrow r(s,a) + \mathbb{E}_{s',a'}\left(Q(s',a') - \alpha \log(\pi(a' \mid s'))\right)$$

   This converges to Q^{π}

Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor

Soft Policy Iteration

Soft policy iteration iterates between two steps:

1. Soft policy evaluation: Fix policy, apply soft Bellman backup operator till convergence

$$Q(s,a) \leftarrow r(s,a) + \mathbb{E}_{s',a'}\left(Q(s',a') - \alpha \log(\pi(a' \mid s'))\right)$$

   This converges to Q^{π}

2. Soft policy improvement: Update the policy:

$$\pi' = \arg\min_{\pi_k \in \Pi} D_{KL}\left(\pi_k(\cdot \mid s_t) \,||\, \frac{\exp(Q^{\pi}(s_t, \cdot))}{Z^{\pi}(s_t)}\right)$$

Leads to a sequence of policies with monotonically increasing soft q values

Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor

Policy Improvement

$$\pi_{new} = \arg\min_{\pi' \in \Pi} D_{KL}\left(\pi'(\cdot \mid s_t) \mid\mid \frac{\exp(Q^{\pi_{\text{old}}}(s_t, \cdot))}{Z^{\pi_{old}}(s_t)}\right)$$

We need to show that this results in a better policy, i.e.,

$$Q^{\pi_{new}}(s_t, a_t) \geq Q^{\pi_{old}}(s_t, a_t) \forall s_t, a_t \in (\mathcal{S} \times \mathcal{A})$$

Policy Improvement

$$\pi_{new} = \arg\min_{\pi' \in \Pi} D_{KL}\left(\pi'(\,\cdot\,|\,s_t) \,||\, \frac{\exp(Q^{\pi_{\text{old}}}(s_t, \cdot\,))}{Z^{\pi_{old}}(s_t)} \right) = \arg\min_{\pi' \in \Pi} J_{\pi_{old}}(\pi'(\,\cdot\,|\,s_t))$$

$$J_{\pi_{old}}(\pi_{new}(\,\cdot\,|\,s_t)) \leq J_{\pi_{old}}(\pi_{old}(\,\cdot\,|\,s_t))$$

$$\mathbb{E}_{a_t \sim \pi_{new}}\left[\log \pi_{new}(a_t\,|\,s_t) - Q^{\pi_{old}}(s_t, a_t) + \log Z^{\pi_{old}}(s_t)\right] \leq \mathbb{E}_{a_t \sim \pi_{old}}\left[\log \pi_{old}(a_t\,|\,s_t) - Q^{\pi_{old}}(s_t, a_t) + \log Z^{\pi_{old}}(s_t)\right]$$

$$\mathbb{E}_{a_t \sim \pi_{new}}\left[\log \pi_{new}(a_t\,|\,s_t) - Q^{\pi_{old}}(s_t, a_t)\right] \leq \mathbb{E}_{a_t \sim \pi_{old}}\left[\log \pi_{old}(a_t\,|\,s_t) - Q^{\pi_{old}}(s_t, a_t)\right]$$

$$\mathbb{E}_{a_t \sim \pi_{new}}\left[Q^{\pi_{old}}(s_t, a_t) - \log \pi_{new}(a_t\,|\,s_t)\right] \geq \mathbb{E}_{a_t \sim \pi_{old}}\left[Q^{\pi_{old}}(s_t, a_t) - \log \pi_{old}(a_t\,|\,s_t)\right]$$

Remember: $V(s_t) = \mathbb{E}_{a_t \sim \pi}\left[Q(s_t, a_t) - \alpha \log \pi(a_t\,|\,s_t)\right]$

$$\mathbb{E}_{a_t \sim \pi_{new}}\left[Q^{\pi_{old}}(s_t, a_t) - \log \pi_{new}(a_t\,|\,s_t)\right] \geq V^{\pi_{old}}(s_t)$$

Policy Improvement

$$\mathbb{E}_{a_t \sim \pi_{new}} Q^{\pi_{old}}(s_t, a_t) - \log \pi_{new}(a_t \mid s_t) \geq V^{\pi_{old}}(s_t)$$

$$\begin{aligned}
Q^{\pi_{old}}(s_t, a_t) &= r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho}[V^{\pi_{old}}(s_{t+1})] \\
&\leq r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim \rho}\left[\mathbb{E}_{a_{t+1} \sim \pi_{new}}\left[Q^{\pi_{old}}(s_{t+1}, a_{t+1}) - \log \pi_{new}(a_{t+1} \mid s_{t+1})\right]\right] \\
&\vdots \\
&\leq Q^{\pi_{new}}(s_t, a_t)
\end{aligned}$$

Soft Policy Iteration

Soft policy iteration iterates between two steps:

1. Soft policy evaluation: Fix policy, apply soft Bellman backup operator till convergence

$$Q(s,a) \leftarrow r(s,a) + \mathbb{E}_{s',a'}\left(Q(s',a') - \alpha \log(\pi(a' \mid s'))\right)$$

   This converges to Q^{π}

2. Soft policy improvement: Update the policy:

$$\pi' = \arg\min_{\pi_k \in \Pi} D_{KL}\left(\pi_k(\cdot \mid s_t) \,||\, \frac{\exp(Q^{\pi}(s_t, \cdot))}{Z^{\pi}(s_t)}\right)$$

Leads to a sequence of policies with monotonically increasing soft q values
This so far concerns tabular methods. Next we will use function approximations for policy and action values

Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor

Soft Policy Iteration - Approximation

Use function approximations for policy and action value functions: $\pi_\phi(a_t \mid s_t)$ $Q_\theta(s_t)$

Soft Policy Iteration - Approximation

Use function approximations for policy and action value functions: $\pi_\phi(a_t \mid s_t)$ $Q_\theta(s_t)$

1. Learning the state-action value function:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim D}\left[\frac{1}{2}\left(Q_\theta(s_t,a_t) - \hat{Q}(s_t,a_t)\right)^2\right]$$

Soft Policy Iteration - Approximation

Use function approximations for policy and action value functions: $\pi_\phi(a_t \mid s_t)$ $Q_\theta(s_t)$

1. Learning the state-action value function:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim D}\left[\frac{1}{2}\left(Q_\theta(s_t,a_t) - \hat{Q}(s_t,a_t)\right)^2\right]$$

Semi-gradient method:

$$\hat{\nabla}_\theta J_Q(\theta) = \nabla_\theta Q_\theta(\mathbf{a}_t,\mathbf{s}_t)\left(Q_\theta(\mathbf{s}_t,\mathbf{a}_t) - \left(r(\mathbf{s}_t,\mathbf{a}_t) + \gamma\left(Q_{\bar{\theta}}(\mathbf{s}_{t+1},\mathbf{a}_{t+1}) - \alpha\log\left(\pi_\phi(\mathbf{a}_{t+1}|\mathbf{s}_{t+1})\right)\right)\right)\right)$$

TD target

Soft Policy Iteration - Approximation

Use function approximations for policy and action value functions: $\pi_\phi(a_t \mid s_t)$ $Q_\theta(s_t)$

2. Learning the policy: $$J_\pi(\phi) = \mathbb{E}_{s_t \in D} D_{KL}\left(\pi_\phi(\cdot \mid s_t) \| \frac{\exp\left(\frac{1}{\alpha} \mathrm{Q}_\theta(\mathrm{s_t}, \cdot)\right)}{Z_\theta(s_t)} \right)$$

$$\nabla_\phi J_\pi(\phi) = \nabla_\phi \mathbb{E}_{s_t \in D} \mathbb{E}_{a_t \sim \pi_\phi(a \mid s_t)} \log \frac{\pi_\phi(a_t \mid s_t)}{\frac{\exp\left(\frac{1}{\alpha} Q_\theta(s_t, \cdot)\right)}{Z_\theta(s_t)}}$$

$$\nabla_\phi J_\pi(\phi) = \nabla_\phi \mathbb{E}_{s_t \in D} \mathbb{E}_{a_t \sim \pi_\phi(a \mid s_t)} \left[\log(\pi_\phi(a_t \mid s_t)) + \log(Z_\theta(s_t)) - \frac{1}{\alpha} Q_\theta(s_t, a_t) \right]$$

But $Z_\theta(s_t) = \int_{\mathcal{A}} \exp(Q_\theta(s_t, a_t)) da_t$ is independent of ϕ

$$\nabla_\phi J_\pi(\phi) = \nabla_\phi \mathbb{E}_{s_t \in D} \mathbb{E}_{a_t \sim \pi_\phi(a \mid s_t)} \left[\log(\pi_\phi(a_t \mid s_t)) - \frac{1}{a} Q_\theta(s_t, a_t) \right]$$

Soft Policy Iteration - Approximation

Use function approximations for policy and action value functions: $\pi_\phi(a_t \mid s_t)$ $Q_\theta(s_t)$

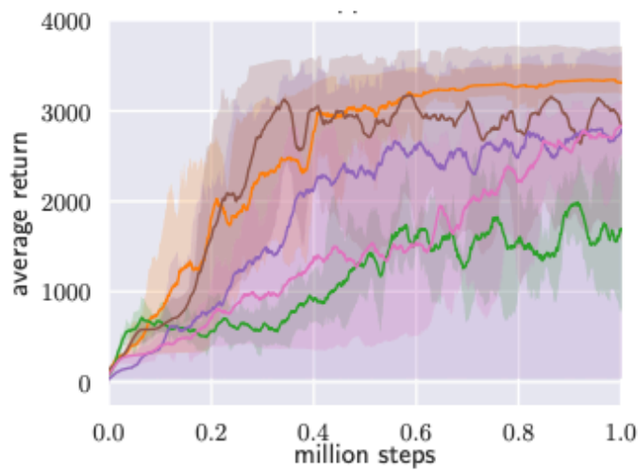
2. Learning the policy: $$J_\pi(\phi) = \mathbb{E}_{s_t \in D} D_{KL}\left(\pi_\phi(\cdot \mid s_t) \| \frac{\exp\left(\frac{1}{\alpha} \mathrm{Q}_\theta(\mathrm{s_t}, \cdot)\right)}{Z_\theta(s_t)}\right)$$

$$\nabla_\phi J_\pi(\phi) = \nabla_\phi \mathbb{E}_{s_t \in D} \mathbb{E}_{a_t \sim \pi_\phi(a \mid s_t)} \left[\log(\pi_\phi(a_t \mid s_t)) - \frac{1}{a} Q_\theta(s_t, a_t)\right]$$

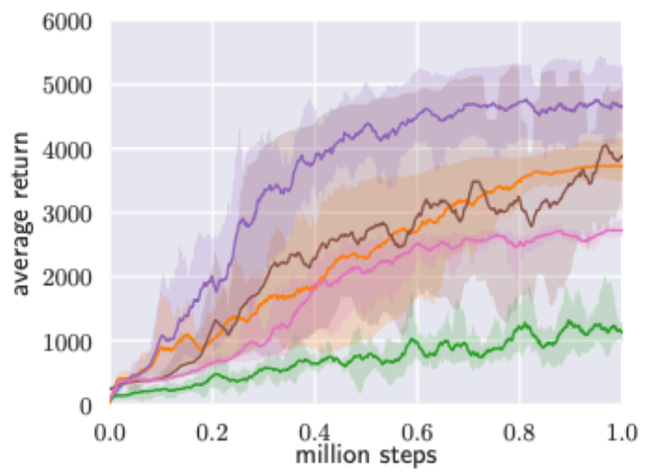
Reparametrization trick: The policy becomes a deterministic function of Gaussian random variables (fixed Gaussian distribution):

$$a_t = f_\phi(s_t, \epsilon) = \mu_\phi(s_t) + \epsilon \Sigma_\phi(s_t), \quad \epsilon \sim \mathcal{N}(0, I)$$

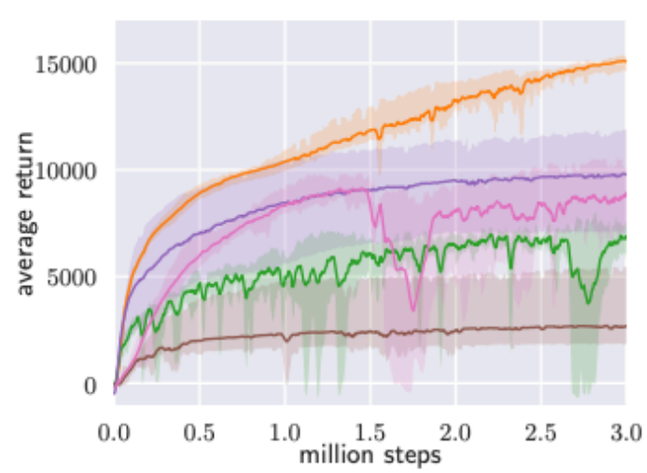
$$\begin{aligned}
\nabla_\phi J_\pi(\phi) &= \nabla_\phi \mathbb{E}_{s_t \in D, \epsilon \sim \mathcal{N}(0,I)} \left[\log(\pi_\phi(a_t \mid s_t)) - \frac{1}{a} Q_\theta(s_t, a_t)\right] \\
&= \mathbb{E}_{s_t \in D, \epsilon \sim \mathcal{N}(0,I)} \nabla_\phi \left[\log(\pi_\phi(a_t \mid s_t)) - \frac{1}{a} Q_\theta(s_t, a_t)\right] \\
&= \mathbb{E}_{s_t \in D, \epsilon \sim \mathcal{N}(0,I)} \nabla_{a_t} \left[\log(\pi_\phi(a_t \mid s_t)) - \frac{1}{a} Q_\theta(s_t, a_t)\right] \nabla_\phi a_t
\end{aligned}$$



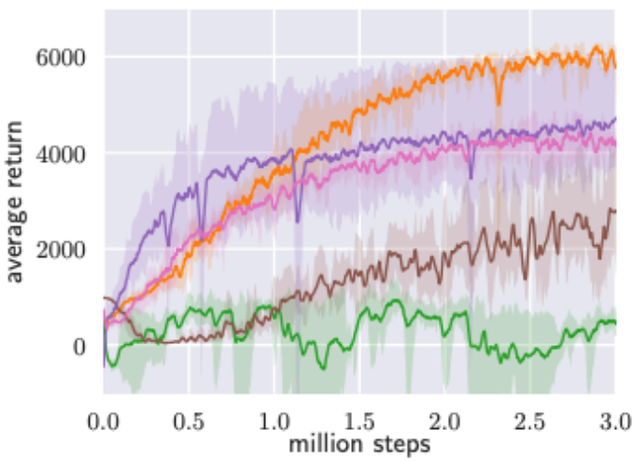
(a) Hopper-v1



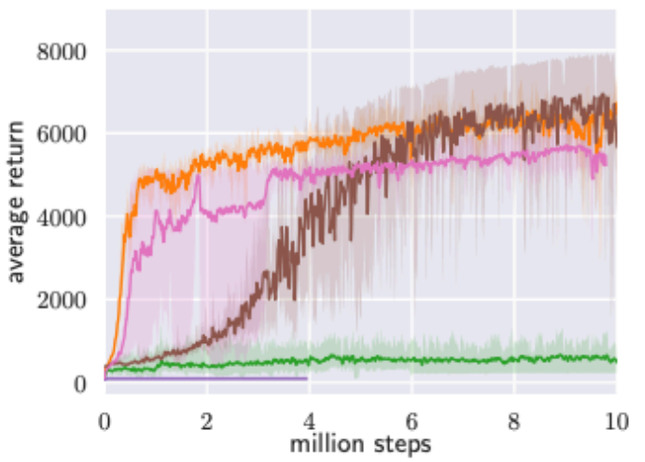
(b) Walker2d-v1



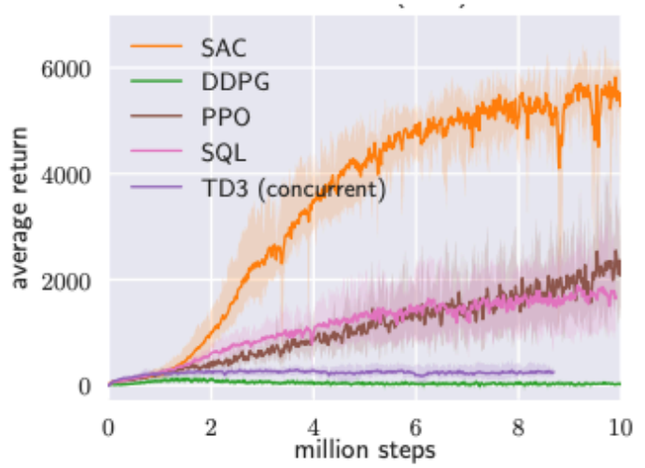
(c) HalfCheetah-v1



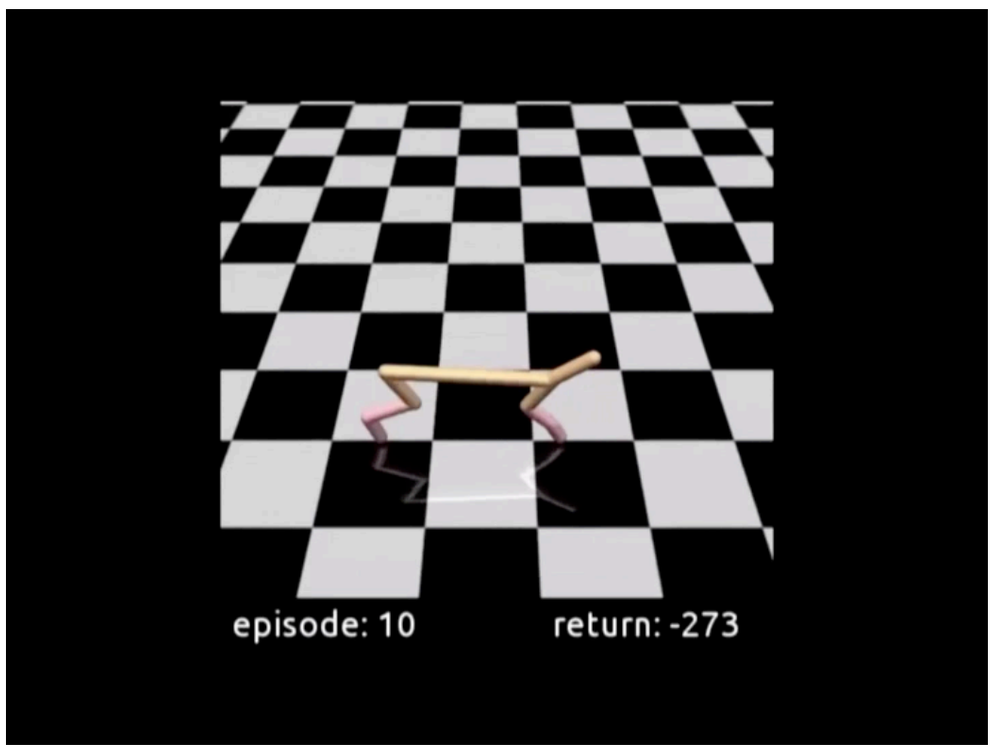
(d) Ant-v1



(e) Humanoid-v1

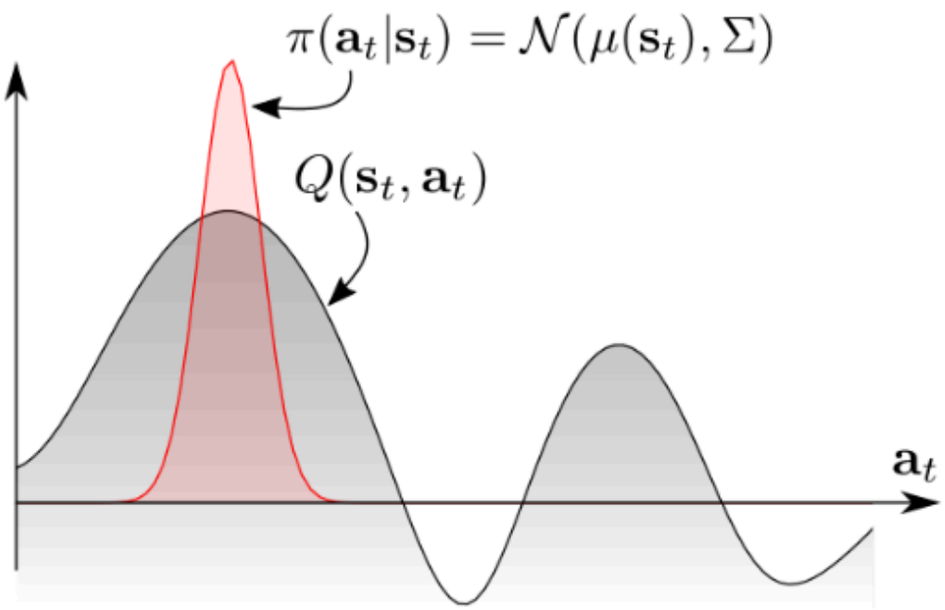


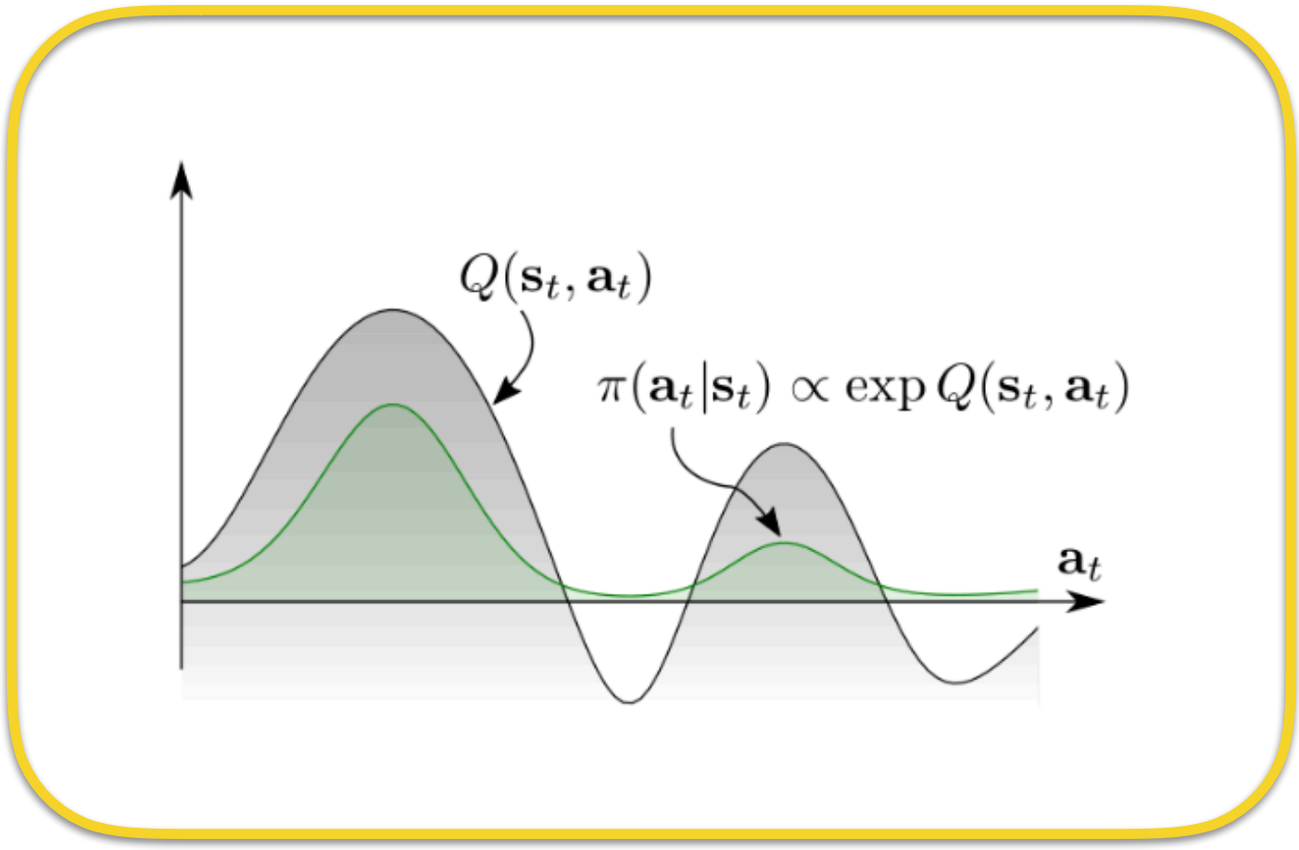
(f) Humanoid (rllab)



Energy-based policies

We need an expressive parametrization for our policy to be able to capture the different modes of our soft Q function.

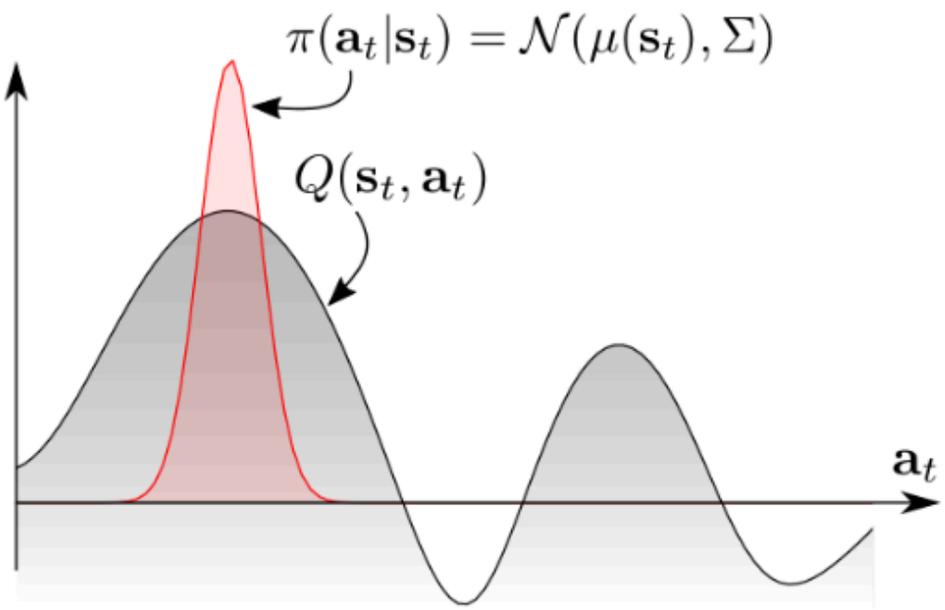


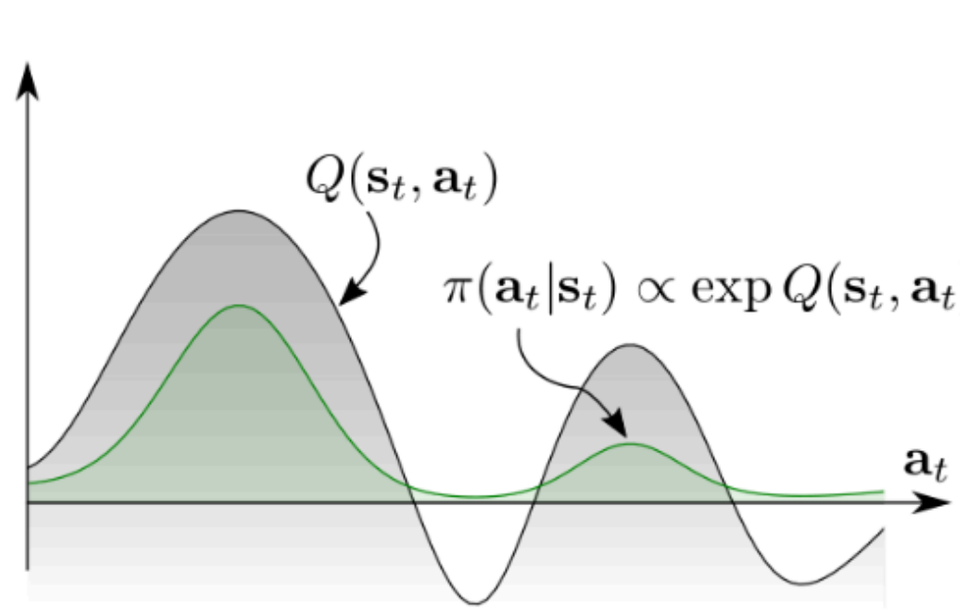


Can we go beyond gaussian and capture multimodal action distributions?

Energy-based policies

We need an expressive parametrization for our policy to be able to capture the different modes of our soft Q function.

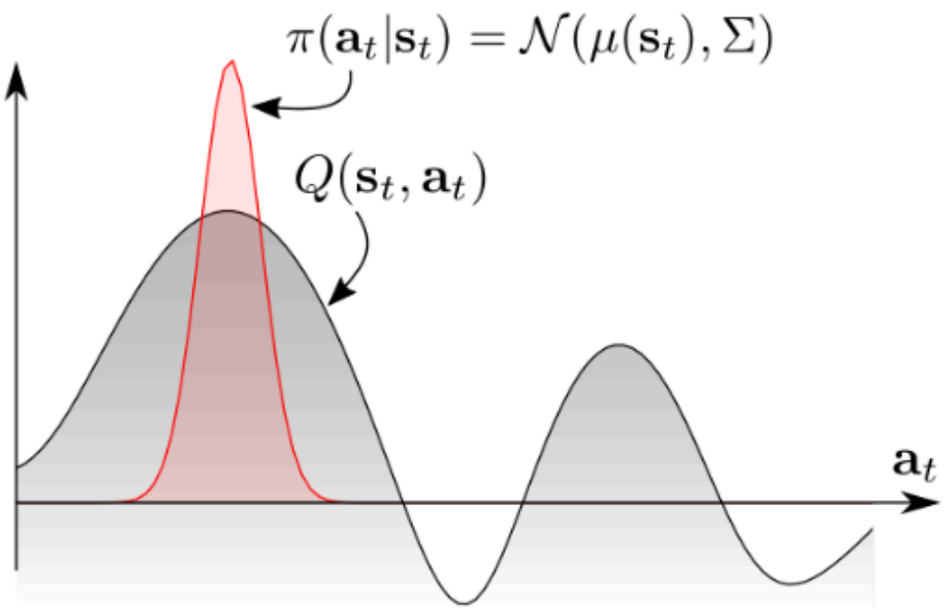


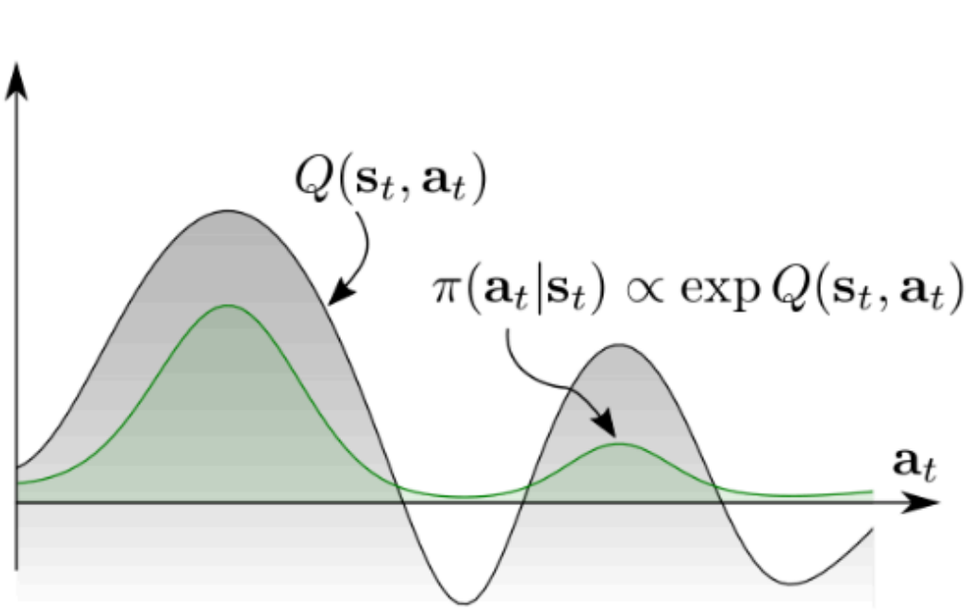


We need to be able to sample from $\exp(\frac{1}{\alpha}Q^{soft}(s_t, a_t))$ which is intractable because the partition function $Z(s_t) = \int_{\mathcal{A}} \exp(Q^{soft}(s_t, a_t))da_t$ is difficult to compute.

Energy-based policies

We need an expressive parametrization for our policy to be able to capture the different modes of our soft Q function.





We need to be able to sample from $\exp(\frac{1}{\alpha}Q^{soft}(s, a))$ which is intractable in general.

We will approximate it using Stein Variational Gradient Descent.

to compute.

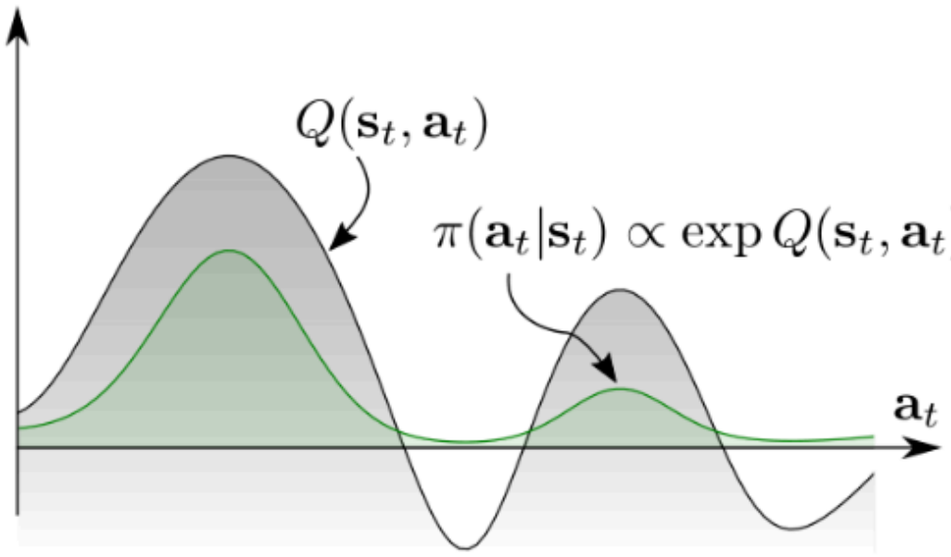
Energy-based policies

We will approximate sampling from the energy based model using Stein Variational Gradient Descent.

Consider the policy to be a state conditioned stochastic network:

$a_t = f^{\phi}(\xi; s_t), \xi \sim \mathcal{N}(0, I)$

We want to estimate parameters ϕ so that we minimize the KL:



$$J_{\pi}(\phi, s_t) = D_{KL}\left(\pi^{\phi}(\,\cdot\,|\,s_t) \| \frac{\exp(Q^{\theta}_{soft}(s_t, \cdot))}{Z^{\theta}(s_t)}\right)$$

This is the same objective we had before but here we do not restrict the policy network to parametrize a Gaussian distribution.

Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm, Liu and Wang 2016

Energy-based policies

Consider the policy to be a state conditioned stochastic network:
$a_t = f^{\phi}(\xi; s_t), \xi \sim \mathcal{N}(0, I)$
We want to estimate parameters ϕ so that we minimize the KL of the action distributions sampled, from the energy based policy.

Algorithm 1 Bayesian Inference via Variational Gradient Descent

Input: A target distribution with density function $p(x)$ and a set of initial particles $\{x_i^0\}_{i=1}^n$.
Output: A set of particles $\{x_i\}_{i=1}^n$ that approximates the target distribution $p(x)$.
for iteration ℓ **do**

$$x_i^{\ell+1} \leftarrow x_i^{\ell} + \epsilon_\ell \hat{\phi}^*(x_i^{\ell}) \quad \text{where} \quad \hat{\phi}^*(x) = \frac{1}{n}\sum_{j=1}^{n}\left[k(x_j^{\ell}, x)\nabla_{x_j^{\ell}} \log p(x_j^{\ell}) + \nabla_{x_j^{\ell}} k(x_j^{\ell}, x)\right],$$

where ϵ_ℓ is the step size at the ℓ-th iteration.
end for

$$\Delta f^{\phi}(\,\cdot\,; s_t) = \frac{1}{n}\sum_{i=1}^{n}\left[\kappa(a_t^i, f^{\phi}(\,\cdot\,; s_t)\,\nabla_{a'} Q_{soft}^{\theta}(s_t, a')\,|_{a'=a_t^i} + \alpha\,\nabla_{a'}\kappa(a', f^{\phi}(\,\cdot\,; s_t))\,|_{a'=a_t^i}\right]$$

$$\frac{\partial J_\pi(\phi; s_t)}{\partial \phi} \propto \frac{1}{n}\sum_{i=1}^{n}\left[\Delta f^{\phi}(\xi^i; s_t)\frac{\partial f^{\phi}(\xi^i; s_t)}{\partial \phi}\right]$$

Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm, Liu and Wang 2016

Energy-based policies

Consider the policy to be a state conditioned stochastic network:
$a_t = f^{\phi}(\xi; s_t), \xi \sim \mathcal{N}(0,I)$

1. Initialize policy network
2. Collect experience
3. Estimate Q_{θ}^{soft} values
4. Update the policy parameters ϕ to minimize the KL with the energy based model $\propto \exp(\frac{1}{a} Q_{\theta}^{soft})$
5. GOTO 2.

Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm, Liu and Wang 2016

Composability of Maximum Entropy Policies

Imagine we want to satisfy two objectives at the same time, e.g., pick an object up while avoiding an obstacle. We would learn a policy to maximize the addition of the the corresponding reward functions:

$$r^C(s,a) = \frac{1}{C}\sum_{i=1}^{C} r_i(s,a)$$

MaxEnt policies permit to obtain the resulting policy's optimal Q by simply adding the constituent Qs:

$$Q_C^*(s,a) \approx \frac{1}{C}\sum_{i=1}^{C} Q_i^*(s,a)$$

We can theoretically bound the suboptimality of the resulting policy w.r.t. the policy trained under the addition of rewards. We cannot do this for deterministic policies.

Composable Deep Reinforcement Learning for Robotic Manipulation, Haarnoja et al.

Composable Deep Reinforcement Learning for Robotic Manipulation

Tuomas Haarnoja, Vitchyr Pong, Aurick Zhou,
Murtaza Dalal, Pieter Abbeel, and Sergey Levine

Berkeley Artificial Intelligence Research
UC Berkeley

https://www.youtube.com/watch?time_continue=82&v=FmMPHL3TcrE