

Policy-Based Reinforcement Learning

Shusen Wang

Policy Function Approximation

Policy Function $\pi(a|s)$

- Policy function $\pi(a|s)$ is a probability density function (PDF).
- It takes state s as input.
- It output the probabilities for all the actions, e.g.,

$$\pi(\text{left}|s) = 0.2,$$

$$\pi(\text{right}|s) = 0.1,$$

$$\pi(\text{up}|s) = 0.7.$$

- The agent performs an action a random drawn from the distribution.

Can we directly learn a policy function $\pi(\textcolor{red}{a}|\textcolor{green}{s})$?

- If there are only a few states and actions, then yes, we can.
- Draw a table (matrix) and learn the entries.

	Action a_1	Action a_2	Action a_3	Action a_4	...
State s_1					
State s_2					
State s_3					
⋮					

Can we directly learn a policy function $\pi(\textcolor{red}{a}|\textcolor{green}{s})$?

- If there are only a few states and actions, then yes, we can.
- Draw a table (matrix) and learn the entries.
- What if there are too many (or infinite) states or actions?

	Action a_1	Action a_2	Action a_3	Action a_4	...
State s_1					
State s_2					
State s_3					
⋮					

Policy Network $\pi(a|s; \theta)$

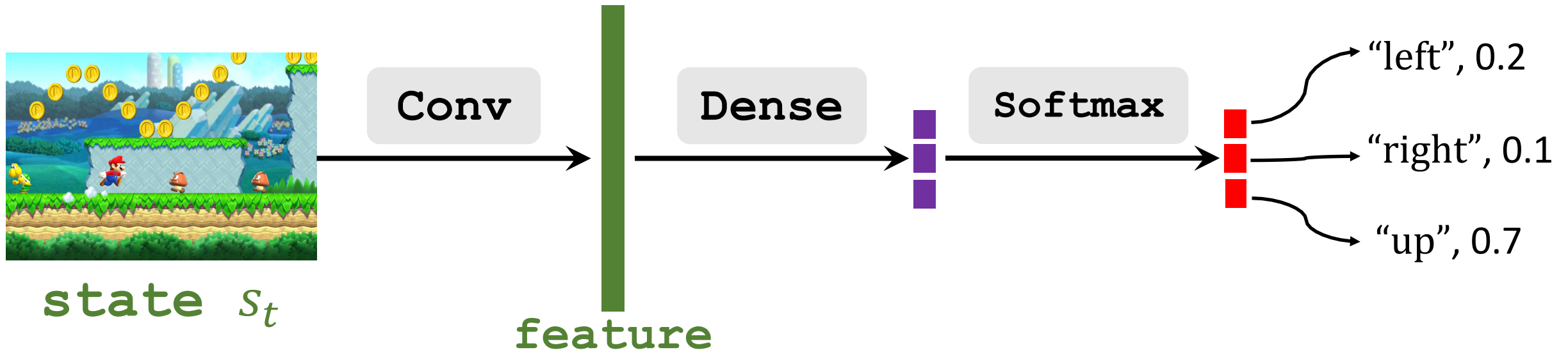
Policy network: Use a neural net to approximate $\pi(a|s)$.

- Use policy network $\pi(a|s; \theta)$ to approximate $\pi(a|s)$.
- θ : trainable parameters of the neural net.

Policy Network $\pi(a|s; \theta)$

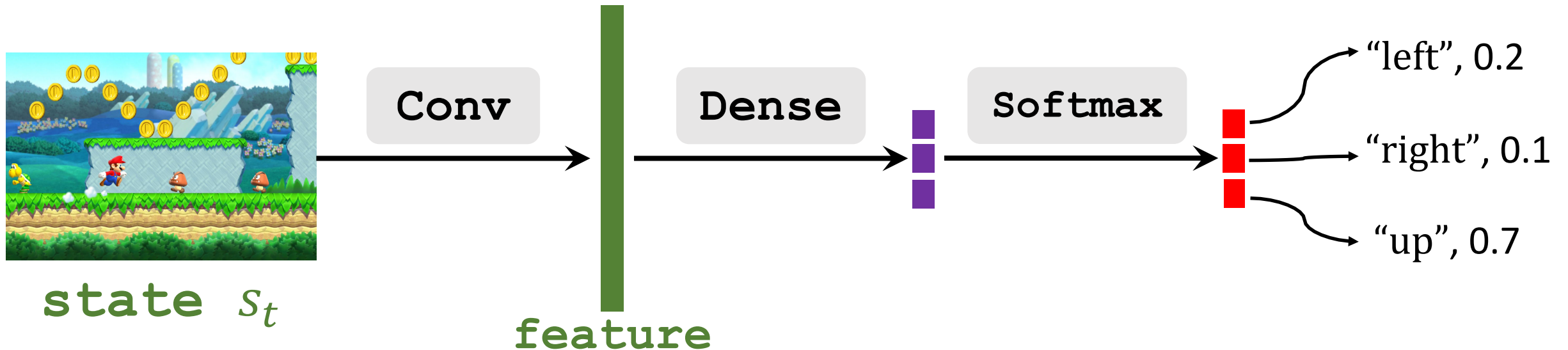
Policy network: Use a neural net to approximate $\pi(a|s)$.

- Use policy network $\pi(a|s; \theta)$ to approximate $\pi(a|s)$.
- θ : trainable parameters of the neural net.



Policy Network $\pi(a|s; \theta)$

- $\sum_{a \in \mathcal{A}} \pi(a|s; \theta) = 1$.
- Here, $\mathcal{A} = \{\text{"left"}, \text{"right"}, \text{"up"}\}$ is the set all actions.
- That is why we use softmax activation.



State-Value Function Approximation

Action-Value Function

Definition: Discounted return.

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \dots$



- The return depends on actions $A_t, A_{t+1}, A_{t+2}, \dots$ and states $S_t, S_{t+1}, S_{t+2}, \dots$
- Actions are random: $\mathbb{P}[A = a \mid S = s] = \pi(a|s)$. (Policy function.)
- States are random: $\mathbb{P}[S' = s' \mid S = s, A = a] = p(s'|s, a)$. (State transition.)


Action-Value Function

Definition: Discounted return.

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \dots$

Definition: Action-value function.

- $Q_\pi(s_t, a_t) = \mathbb{E} [U_t | S_t = s_t, A_t = a_t].$



The expectation is taken w.r.t.
actions $A_{t+1}, A_{t+2}, A_{t+3}, \dots$
and states $S_{t+1}, S_{t+2}, S_{t+3}, \dots$

State-Value Function

Definition: Discounted return.

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \dots$

Definition: Action-value function.

- $Q_\pi(s_t, a_t) = \mathbb{E} [U_t | S_t = s_t, A_t = a_t].$

Definition: State-value function.

- $V_\pi(s_t) = \mathbb{E}_A [Q_\pi(s_t, A)]$



Integrate out action $A \sim \pi(\cdot | s_t).$

State-Value Function

Definition: Discounted return.

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \dots$

Definition: Action-value function.

- $Q_\pi(s_t, a_t) = \mathbb{E} [U_t | S_t = s_t, A_t = a_t].$

Definition: State-value function.

- $V_\pi(s_t) = \mathbb{E}_A [Q_\pi(s_t, A)] = \sum_a \pi(a | s_t) \cdot Q_\pi(s_t, a).$



Integrate out action $A \sim \pi(\cdot | s_t).$

Policy-Based Reinforcement Learning

Definition: State-value function.

- $V_{\pi}(s_t) = \mathbb{E}_A [Q_{\pi}(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_{\pi}(s_t, a).$

Policy-Based Reinforcement Learning

Definition: State-value function.

- $V_{\pi}(s_t) = \mathbb{E}_A [Q_{\pi}(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_{\pi}(s_t, a).$

Approximate state-value function.

- Approximate policy function $\pi(a|s_t)$ by policy network $\pi(a|s_t; \theta).$

Policy-Based Reinforcement Learning

Definition: State-value function.

- $V_{\pi}(s_t) = \mathbb{E}_A [Q_{\pi}(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_{\pi}(s_t, a).$

Approximate state-value function.

- Approximate policy function $\pi(a|s_t)$ by policy network $\pi(a|s_t; \theta).$
- Approximate value function $V_{\pi}(s_t)$ by:

$$V(s_t; \theta) = \sum_a \pi(a|s_t; \theta) \cdot Q_{\pi}(s_t, a).$$

Policy-Based Reinforcement Learning

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy-based learning: Learn θ that maximizes $J(\theta) = \mathbb{E}_s[V(S; \theta)].$

Policy-Based Reinforcement Learning

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy-based learning: Learn θ that maximizes $J(\theta) = \mathbb{E}_s[V(S; \theta)].$

How to improve θ ? Policy gradient ascent!

- Observe state s .

- Update policy by: $\theta \leftarrow \theta + \beta \cdot \frac{\partial V(s; \theta)}{\partial \theta}$

Policy gradient

Policy Gradient

Reference

- Sutton and others: [Policy gradient methods for reinforcement learning with function approximation](#). In *NIPS*, 2000.

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $\frac{\partial V(s; \theta)}{\partial \theta}$

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\frac{\partial V(s; \theta)}{\partial \theta} = \frac{\partial \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a)}{\partial \theta}$$

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\frac{\partial V(s; \theta)}{\partial \theta} = \frac{\partial \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a)}{\partial \theta}$$
$$= \sum_a \frac{\partial \pi(a|s; \theta) \cdot Q_\pi(s, a)}{\partial \theta}$$

← Push derivative inside the summation

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\begin{aligned} \frac{\partial V(s; \theta)}{\partial \theta} &= \frac{\partial \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a)}{\partial \theta} \\ &= \sum_a \frac{\partial \pi(a|s; \theta) \cdot Q_\pi(s, a)}{\partial \theta} \\ &= \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \end{aligned}$$

← Pretend Q_π is independent of θ .
(It may not be true.)

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $\frac{\partial V(s; \theta)}{\partial \theta} = \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a)$

Policy Gradient: Form 1

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $\frac{\partial V(s; \theta)}{\partial \theta} = \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a)$

Policy Gradient: Form 1

Note: This derivation is **over-simplified** and **not rigorous**.

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\begin{aligned} \frac{\partial V(s; \theta)}{\partial \theta} &= \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} Q_\pi(s, a) \\ &= \sum_a \pi(a|s; \theta) \cdot \frac{\partial \log \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \end{aligned}$$

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\begin{aligned} \frac{\partial V(s; \theta)}{\partial \theta} &= \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \\ &= \sum_a \pi(a|s; \theta) \cdot \boxed{\frac{\partial \log \pi(a|s; \theta)}{\partial \theta}} \cdot Q_\pi(s, a) \end{aligned}$$

- Chain rule: $\frac{\partial \log[\pi(\theta)]}{\partial \theta} = \frac{1}{\pi(\theta)} \cdot \frac{\partial \pi(\theta)}{\partial \theta}.$

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\begin{aligned} \frac{\partial V(s; \theta)}{\partial \theta} &= \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \\ &= \sum_a \pi(a|s; \theta) \cdot \frac{\partial \log \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \end{aligned}$$

- Chain rule: $\frac{\partial \log[\pi(\theta)]}{\partial \theta} = \frac{1}{\pi(\theta)} \cdot \frac{\partial \pi(\theta)}{\partial \theta}.$
- $\rightarrow \pi(\theta) \cdot \frac{\partial \log[\pi(\theta)]}{\partial \theta} = \pi(\theta) \cdot \frac{1}{\pi(\theta)} \cdot \frac{\partial \pi(\theta)}{\partial \theta}$

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\begin{aligned} \frac{\partial V(s; \theta)}{\partial \theta} &= \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \\ &= \sum_a \pi(a|s; \theta) \cdot \frac{\partial \log \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \end{aligned}$$

- Chain rule: $\frac{\partial \log[\pi(\theta)]}{\partial \theta} = \frac{1}{\pi(\theta)} \cdot \frac{\partial \pi(\theta)}{\partial \theta}.$
- $\rightarrow \pi(\theta) \cdot \frac{\partial \log[\pi(\theta)]}{\partial \theta} = \cancel{\pi(\theta)} \cdot \cancel{\frac{1}{\pi(\theta)}} \cdot \frac{\partial \pi(\theta)}{\partial \theta}.$

Policy Gradient

Definition: Approximate state-value function.

- $V(s; \theta) = \sum_a \pi(a|s; \theta) \cdot Q_\pi(s, a).$

Policy gradient: Derivative of $V(s; \theta)$ w.r.t. θ .

- $$\begin{aligned} \frac{\partial V(s; \theta)}{\partial \theta} &= \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} Q_\pi(s, a) \\ &= \sum_a \pi(a|s; \theta) \cdot \frac{\partial \log \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a) \end{aligned}$$

- Chain rule: $\frac{\partial \log[\pi(\theta)]}{\partial \theta} = \frac{1}{\pi(\theta)} \cdot \frac{\partial \pi(\theta)}{\partial \theta}.$
- $\rightarrow \pi(\theta) \cdot \frac{\partial \log[\pi(\theta)]}{\partial \theta} = \cancel{\pi(\theta)} \cdot \frac{1}{\cancel{\pi(\theta)}} \cdot \frac{\partial \pi(\theta)}{\partial \theta}.$

Policy Gradient

Definition: Approximate state-value function.

- $V(\mathbf{s}; \boldsymbol{\theta}) = \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta}) \cdot Q_{\pi}(\mathbf{s}, \mathbf{a}).$

Policy gradient: Derivative of $V(\mathbf{s}; \boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$.

- $$\begin{aligned} \frac{\partial V(\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \sum_{\mathbf{a}} \frac{\partial \pi(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \mathbf{a}) \\ &= \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta}) \cdot \frac{\partial \log \pi(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \mathbf{a}) \\ &= \mathbb{E}_{\mathbf{A}} \left[\frac{\partial \log \pi(\mathbf{A}|\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \mathbf{A}) \right]. \end{aligned}$$

The expectation is taken w.r.t. the random variable $\mathbf{A} \sim \pi(\cdot | \mathbf{s}; \boldsymbol{\theta})$.

Policy Gradient

Two forms of policy gradient:

- Form 1: $\frac{\partial V(\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{\mathbf{a}} \frac{\partial \pi(\mathbf{a} | \mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \mathbf{a}).$
- Form 2: $\frac{\partial V(\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbb{E}_{\mathbf{A} \sim \pi(\cdot | \mathbf{s}; \boldsymbol{\theta})} \left[\frac{\partial \log \pi(\mathbf{A} | \mathbf{s}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \mathbf{A}) \right].$

Calculate Policy Gradient for Discrete Actions

If the actions are **discrete**, e.g., action space $\mathcal{A} = \{\text{"left"}, \text{"right"}, \text{"up"}\}, \dots$

Use **Form 1**: $\frac{\partial V(s; \theta)}{\partial \theta} = \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_{\pi}(s, a).$

Calculate Policy Gradient for Discrete Actions

If the actions are **discrete**, e.g., action space $\mathcal{A} = \{\text{"left"}, \text{"right"}, \text{"up"}\}, \dots$

Use **Form 1**: $\frac{\partial V(s; \theta)}{\partial \theta} = \sum_a \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a).$

1. Calculate $\mathbf{f}(a, \theta) = \frac{\partial \pi(a|s; \theta)}{\partial \theta} \cdot Q_\pi(s, a)$, for every action $a \in \mathcal{A}$.
2. Policy gradient: $\frac{\partial V(s; \theta)}{\partial \theta} = \mathbf{f}(\text{"left"}, \theta) + \mathbf{f}(\text{"right"}, \theta) + \mathbf{f}(\text{"up"}, \theta).$

If $|\mathcal{A}|$ is big, this approach is costly.

Calculate Policy Gradient for Continuous Actions

If the actions are **continuous**, e.g., action space $\mathcal{A} = [0, 1], \dots$

Use **Form 2**:
$$\frac{\partial V(s; \theta)}{\partial \theta} = \mathbb{E}_{A \sim \pi(\cdot | s; \theta)} \left[\frac{\partial \log \pi(A | s, \theta)}{\partial \theta} \cdot Q_{\pi}(s, A) \right].$$

Calculate Policy Gradient for Continuous Actions

If the actions are **continuous**, e.g., action space $\mathcal{A} = [0, 1], \dots$

Use **Form 2**:
$$\frac{\partial V(s; \theta)}{\partial \theta} = \mathbb{E}_{A \sim \pi(\cdot | s; \theta)} \left[\frac{\partial \log \pi(A | s, \theta)}{\partial \theta} \cdot Q_{\pi}(s, A) \right].$$

1. Randomly sample an action \hat{a} according to the PDF $\pi(\cdot | s; \theta)$.

Calculate Policy Gradient for Continuous Actions

If the actions are **continuous**, e.g., action space $\mathcal{A} = [0, 1], \dots$

Use **Form 2**: $\frac{\partial V(s; \theta)}{\partial \theta} = \mathbb{E}_{A \sim \pi(\cdot | s; \theta)} \left[\frac{\partial \log \pi(A | s, \theta)}{\partial \theta} \cdot Q_{\pi}(s, A) \right].$

1. Randomly sample an action \hat{a} according to the PDF $\pi(\cdot | s; \theta)$.

2. Calculate $\mathbf{g}(\hat{a}, \theta) = \frac{\partial \log \pi(\hat{a} | s; \theta)}{\partial \theta} \cdot Q_{\pi}(s, \hat{a}).$

- By the definition of \mathbf{g} , $\mathbb{E}_A[\mathbf{g}(A, \theta)] = \frac{\partial V(s; \theta)}{\partial \theta}.$
- $\mathbf{g}(\hat{a}, \theta)$ is an unbiased estimate of $\frac{\partial V(s; \theta)}{\partial \theta}.$

Calculate Policy Gradient for Continuous Actions

If the actions are **continuous**, e.g., action space $\mathcal{A} = [0, 1], \dots$

Use **Form 2**:
$$\frac{\partial V(\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbb{E}_{\mathbf{A} \sim \pi(\cdot | \mathbf{s}; \boldsymbol{\theta})} \left[\frac{\partial \log \pi(\mathbf{A} | \mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \mathbf{A}) \right].$$

1. Randomly sample an action $\hat{\mathbf{a}}$ according to the PDF $\pi(\cdot | \mathbf{s}; \boldsymbol{\theta})$.
2. Calculate $\mathbf{g}(\hat{\mathbf{a}}, \boldsymbol{\theta}) = \frac{\partial \log \pi(\hat{\mathbf{a}} | \mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \cdot Q_{\pi}(\mathbf{s}, \hat{\mathbf{a}})$.
3. Use $\mathbf{g}(\hat{\mathbf{a}}, \boldsymbol{\theta})$ as an approximation to the policy gradient $\frac{\partial V(\mathbf{s}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$.

Update **policy network** using **policy gradient**

Algorithm

1. Observe the state s_t .
2. Randomly sample action a_t according to $\pi(\cdot | s_t; \theta_t)$.

Algorithm

1. Observe the state s_t .
2. Randomly sample action a_t according to $\pi(\cdot | s_t; \theta_t)$.
3. Compute $q_t \approx Q_\pi(s_t, a_t)$ (some estimate).
4. Differentiate policy network: $\mathbf{d}_{\theta,t} = \frac{\partial \log \pi(a_t | s_t, \theta)}{\partial \theta} \big|_{\theta=\theta_t}$.

Algorithm

1. Observe the state s_t .
2. Randomly sample action a_t according to $\pi(\cdot | s_t; \theta_t)$.
3. Compute $q_t \approx Q_\pi(s_t, a_t)$ (some estimate).
4. Differentiate policy network: $\mathbf{d}_{\theta,t} = \frac{\partial \log \pi(a_t | s_t, \theta)}{\partial \theta} \big|_{\theta=\theta_t}$.
5. (Approximate) policy gradient: $\mathbf{g}(a_t, \theta_t) = q_t \cdot \mathbf{d}_{\theta,t}$.
6. Update policy network: $\theta_{t+1} = \theta_t + \beta \cdot \mathbf{g}(a_t, \theta_t)$.

Algorithm

1. Observe the state s_t .
2. Randomly sample action a_t according to $\pi(\cdot | s_t; \theta_t)$.
3. Compute $q_t \approx Q_\pi(s_t, a_t)$ (some estimate). **How?**
4. Differentiate policy network: $\mathbf{d}_{\theta,t} = \frac{\partial \log \pi(a_t | s_t, \theta)}{\partial \theta} \big|_{\theta=\theta_t}$.
5. (Approximate) policy gradient: $\mathbf{g}(a_t, \theta_t) = q_t \cdot \mathbf{d}_{\theta,t}$.
6. Update policy network: $\theta_{t+1} = \theta_t + \beta \cdot \mathbf{g}(a_t, \theta_t)$.

Algorithm

1. Observe the state s_t .
2. Randomly sample action a_t according to $\pi(\cdot | s_t; \theta_t)$.
3. Compute $q_t \approx Q_\pi(s_t, a_t)$ (some estimate). **How?**

Option 1: REINFORCE.

- Play the game to the end and generate the trajectory:

$$s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_T, a_T, r_T.$$

- Compute the discounted return $u_t = \sum_{k=t}^T \gamma^{k-t} r_k$, for all t .
- Since $Q_\pi(s_t, a_t) = \mathbb{E}[U_t]$, we can use u_t to approximate $Q_\pi(s_t, a_t)$.
- \Rightarrow Use $q_t = u_t$.

Algorithm

1. Observe the state s_t .
2. Randomly sample action a_t according to $\pi(\cdot | s_t; \theta_t)$.
3. Compute $q_t \approx Q_\pi(s_t, a_t)$ (some estimate). **How?**

Option 2: Approximate Q_π using a neural network.

- This leads to the actor-critic method.

Summary

Policy-Based Learning

- If a good policy function π is known, the agent can be controlled by the policy: randomly sample $a_t \sim \pi(\cdot | s_t)$.
- Approximate policy function $\pi(a|s)$ by **policy network** $\pi(a|s; \theta)$.
- Learn the policy network by **policy gradient algorithm**.
- Policy gradient algorithm learn θ that maximizes $\mathbb{E}_s[V(s; \theta)]$.

Thank you!