强化学习与博弈论

Reinforcement Learning and Game Theory

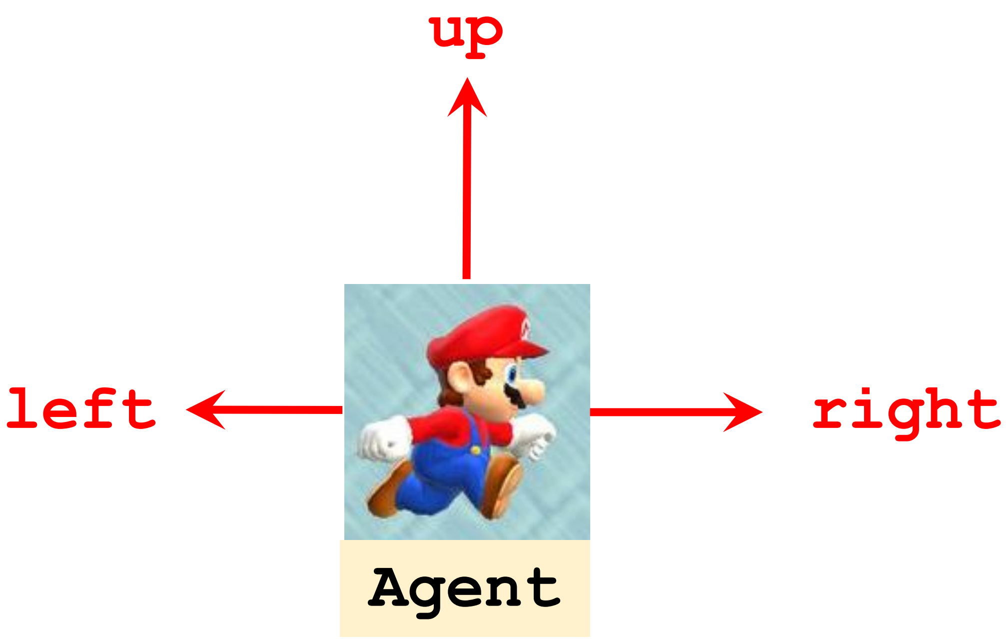
陈 旭

计算机学院

中山大學
SUN YAT-SEN UNIVERSITY

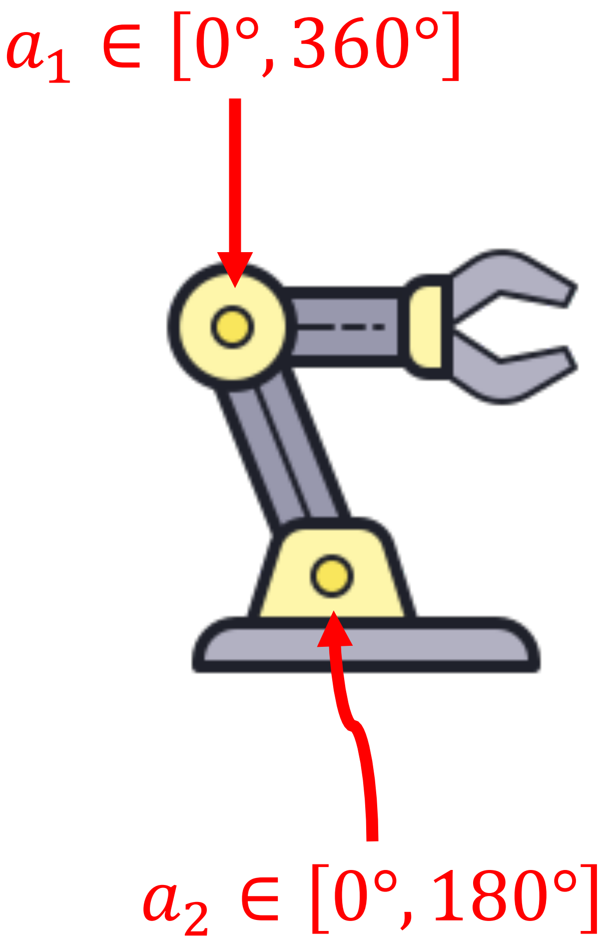
Deterministic Policy Gradient RL

Discrete Action Space



- Action space $\mathcal{A} = \{\text{left}, \text{right}, \text{up}\}$.
- The action space $\mathcal{A}$ is discrete.

Continuous Action Space

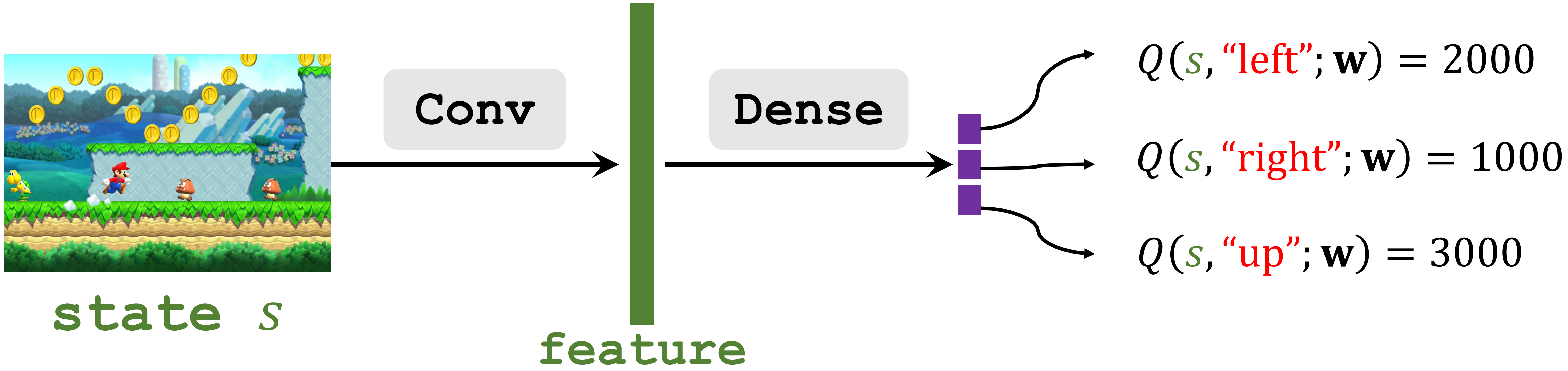


- The action space $\mathcal{A}$ is continuous:

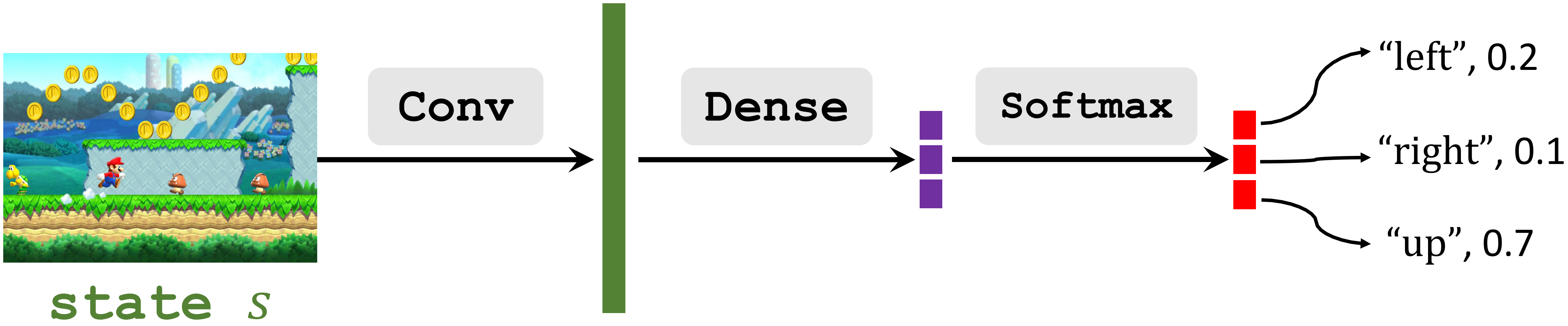
$$\mathcal{A} = [0°, 360°] \times [0°, 180°].$$

- Actions are 2-dim vectors.

DQN for Discrete Action Space



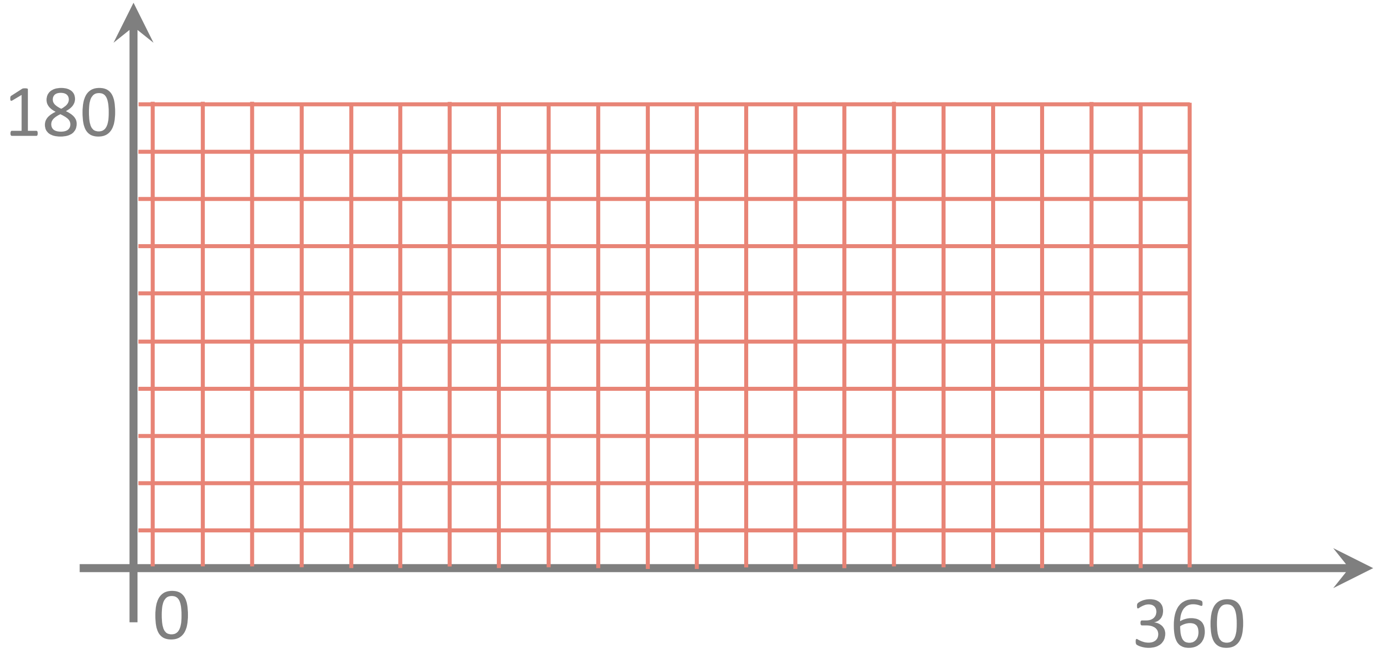
Policy Network for Discrete Action Space



Discretization

Discretization

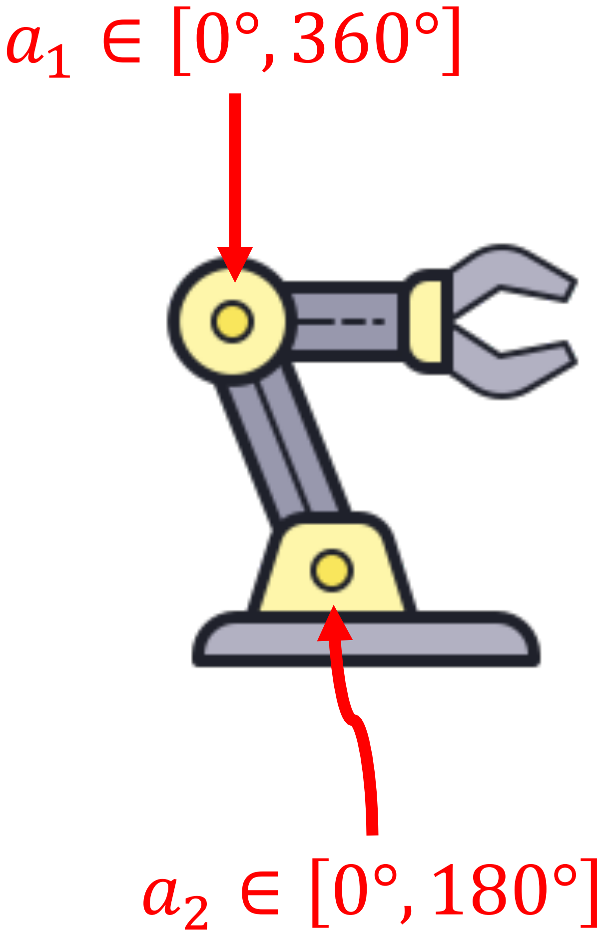
- Discretize the action space. (Draw a grid.)
- Now, the number of actions is the number of grid points.



Discretization

- Discretize the action space. (Draw a grid.)
- Now, the number of actions is the number of grid points.
- Problem: curse of dimensionality.
  - Let d be the degree of freedom.
  - The number of actions grows exponentially with d.

Continuous Action Space



- The action space $\mathcal{A}$ is a subset of $\mathbb{R}^2$.
- The action space $\mathcal{A}$ is continuous:

$$\mathcal{A} = [0°, 360°] \times [0°, 180°].$$

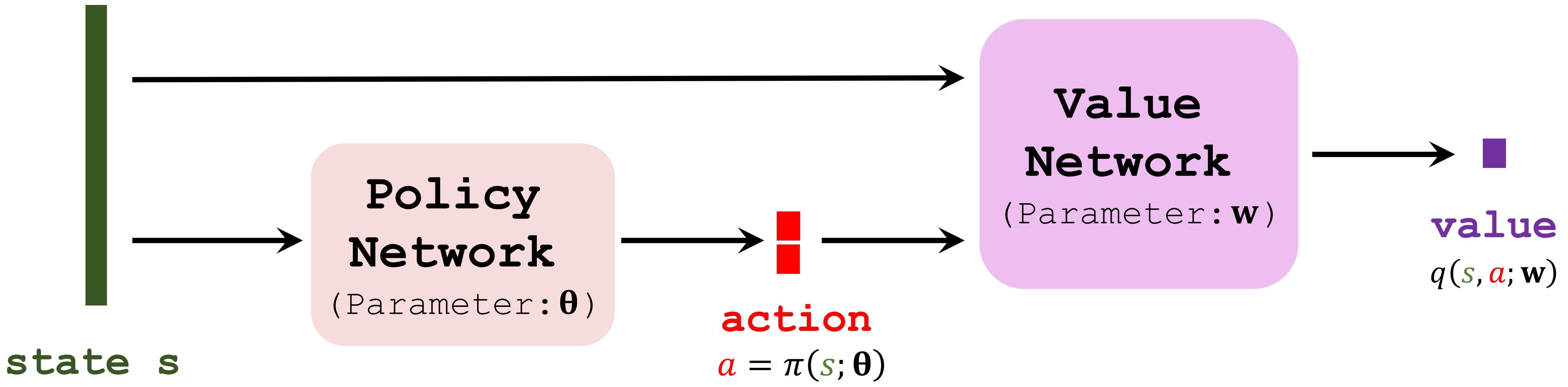
- Actions are 2-dim vectors.

Deterministic Policy Gradient (DPG)

Reference:

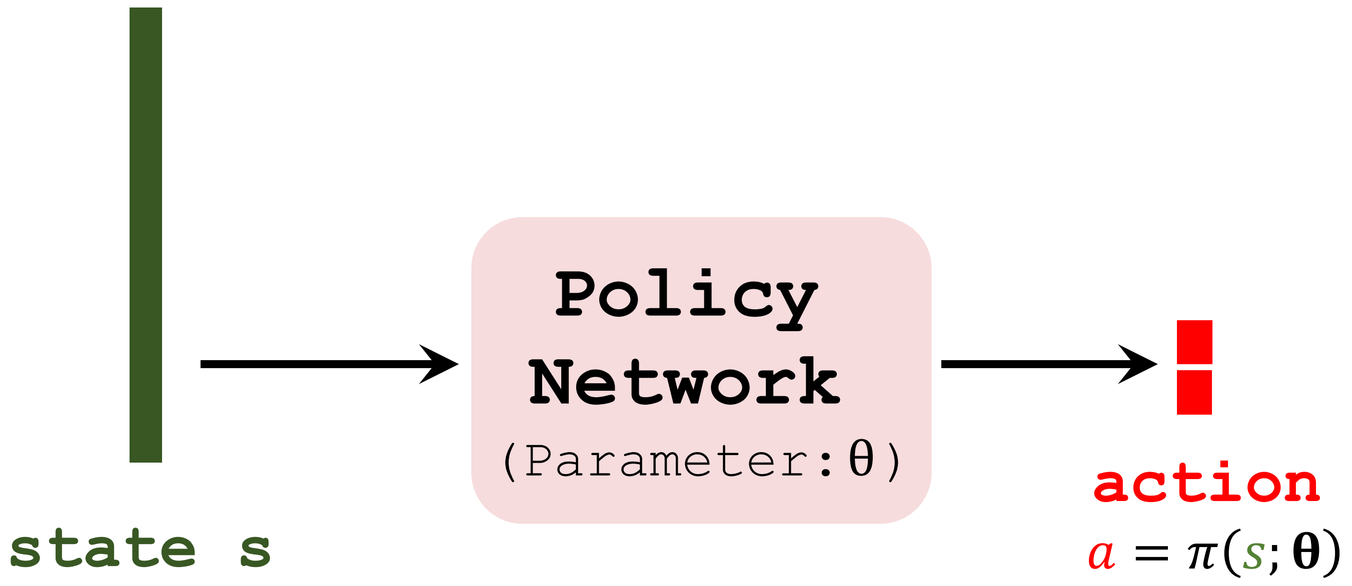
- Silver et al. Deterministic policy gradient algorithms. In *ICML*, 2014.
- Lillicrap et al. Continuous control with deep reinforcement learning. In *ICLR*, 2016.

Deterministic Actor-Critic



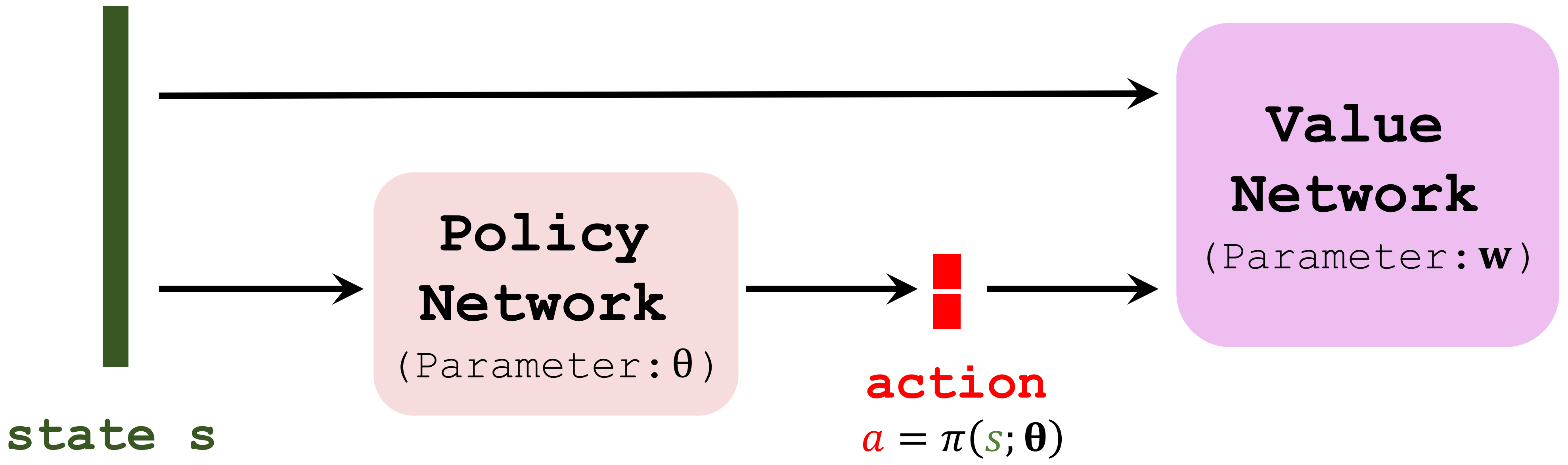
Deterministic Actor-Critic

- Use a deterministic policy network (actor): $a = \pi(s; \boldsymbol{\theta})$.



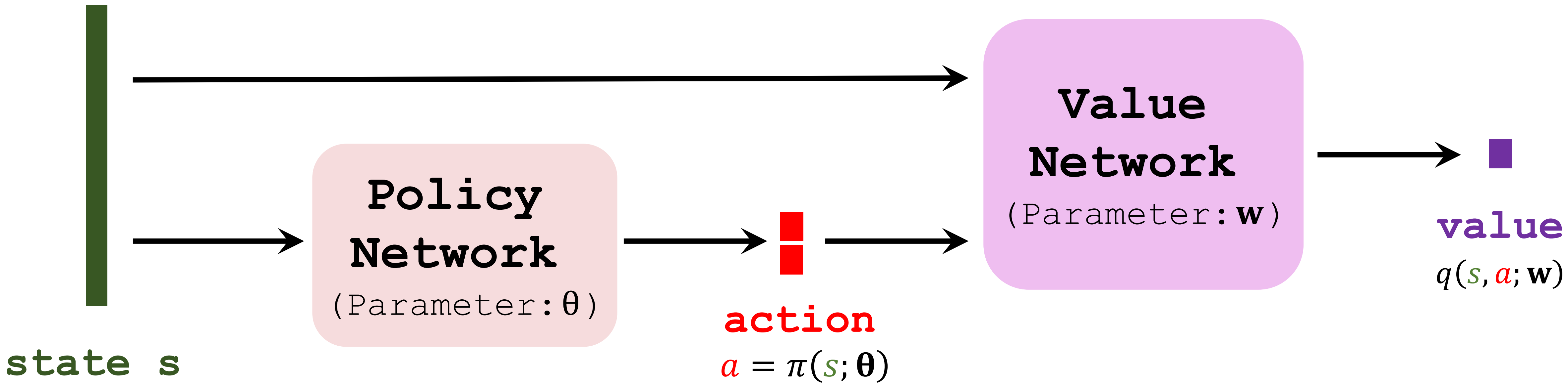
Deterministic Actor-Critic

- Use a deterministic policy network (actor): $a = \pi(s; \boldsymbol{\theta})$.
- Use a value network (critic): $q(s, a; \mathbf{w})$.



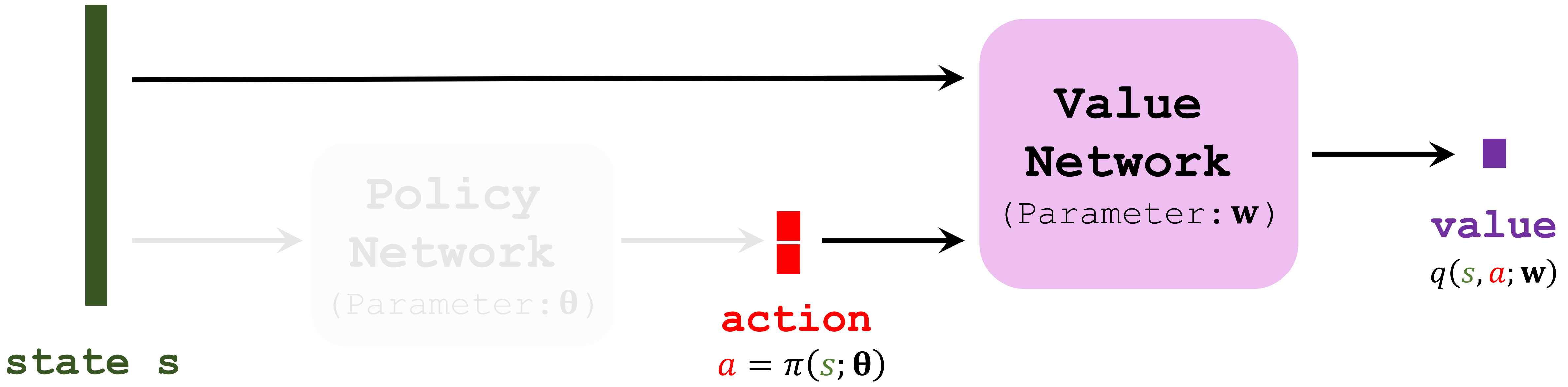
Deterministic Actor-Critic

- Use a deterministic policy network (actor): $a = \pi(s; \boldsymbol{\theta})$.
- Use a value network (critic): $q(s, a; \mathbf{w})$.
- The critic outputs a scalar that evaluates *how good the action* a *is*.



Updating Value Network by TD

- Transition: (s_t, a_t, r_t, s_{t+1}).



Updating Value Network by TD

- Transition: (s_t, a_t, r_t, s_{t+1}).
- Value network makes prediction for time t:

$$q_t = q(s_t, a_t; \mathbf{w}).$$

Updating Value Network by TD

- Transition: (s_t, a_t, r_t, s_{t+1}).
- Value network makes prediction for time t:

$$q_t = q(s_t, a_t; \mathbf{w}).$$

- Value network makes prediction for time $t+1$:

$$q_{t+1} = q(s_{t+1}, a'_{t+1}; \mathbf{w}), \text{ where } a'_{t+1} = \pi(s_{t+1}; \boldsymbol{\theta}).$$

Updating Value Network by TD

- Transition: (s_t, a_t, r_t, s_{t+1}).
- Value network makes prediction for time t:

$$q_t = q(s_t, a_t; \mathbf{w}).$$

- Value network makes prediction for time $t+1$:

$$q_{t+1} = q(s_{t+1}, a'_{t+1}; \mathbf{w}), \text{ where } a'_{t+1} = \pi(s_{t+1}; \boldsymbol{\theta}).$$

- TD error: $\delta_t = q_t - \underbrace{(r_t + \gamma \cdot q_{t+1})}_{\text{TD Target}}.$

Updating Value Network by TD

- Transition: (s_t, a_t, r_t, s_{t+1}).
- Value network makes prediction for time t:

$$q_t = q(s_t, a_t; \mathbf{w}).$$

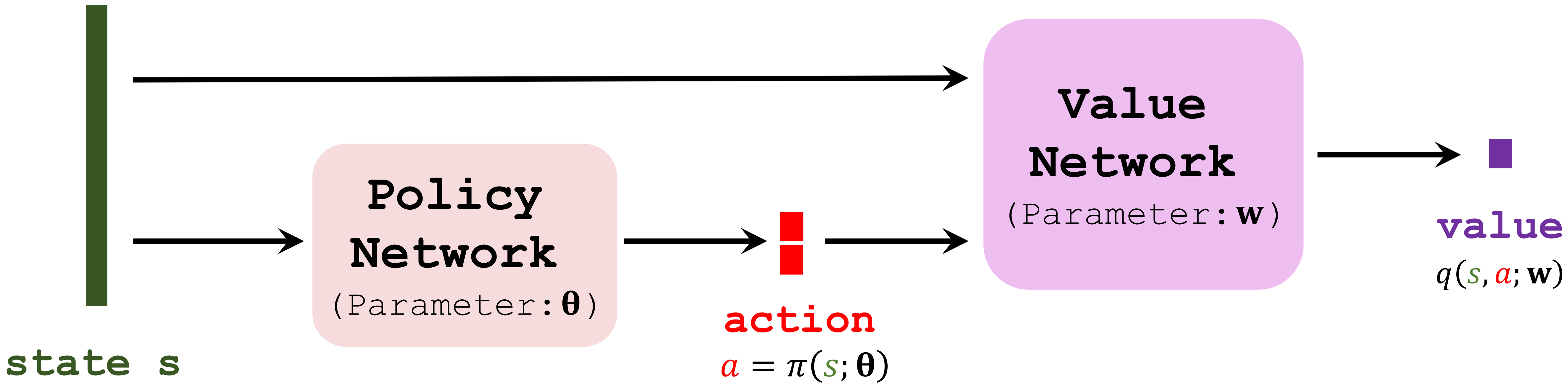
- Value network makes prediction for time $t+1$:

$$q_{t+1} = q(s_{t+1}, a'_{t+1}; \mathbf{w}), \text{ where } a'_{t+1} = \pi(s_{t+1}; \boldsymbol{\theta}).$$

- TD error: $\delta_t = q_t - (r_t + \gamma \cdot q_{t+1})$.
- Update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \delta_t \cdot \frac{\partial\, q(s_t, a_t; \mathbf{w})}{\partial \mathbf{w}}$.

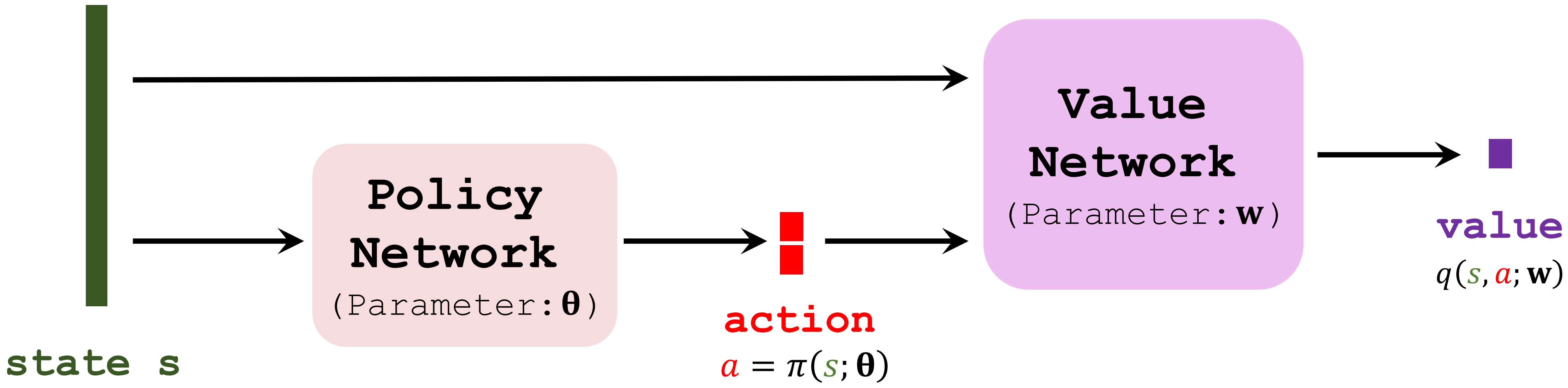
Updating Policy Network by DPG

- The critic $q(s, a; \mathbf{w})$ evaluates how good the action a is.



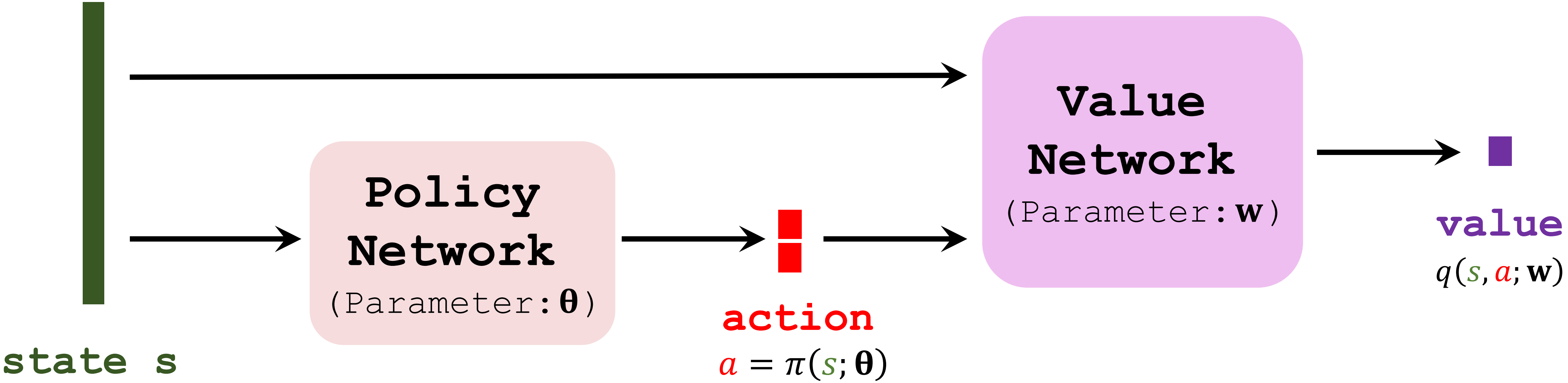
Updating Policy Network by DPG

- The critic $q(s, a; \mathbf{w})$ evaluates how good the action a is.
- Improve $\boldsymbol{\theta}$ so that the critic believes $a = \pi(s; \boldsymbol{\theta})$ is better.
- Update $\boldsymbol{\theta}$ so that $q(s, a; \mathbf{w}) = q(s, \pi(s; \boldsymbol{\theta}); \mathbf{w})$ increases.



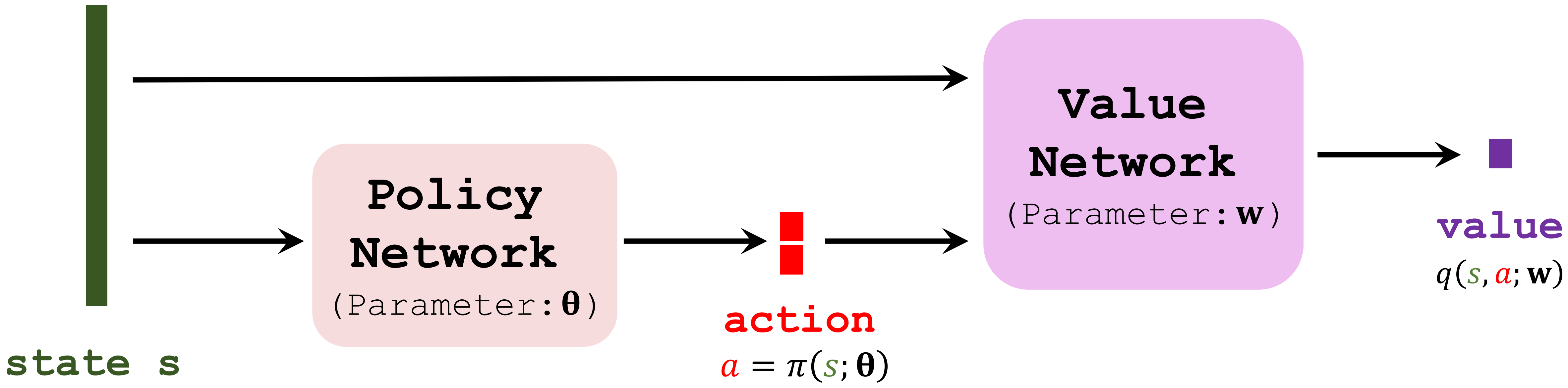
Updating Policy Network by DPG

- **Goal:** Increasing $q(s, a; \mathbf{w})$, where $a = \pi(s; \boldsymbol{\theta})$.



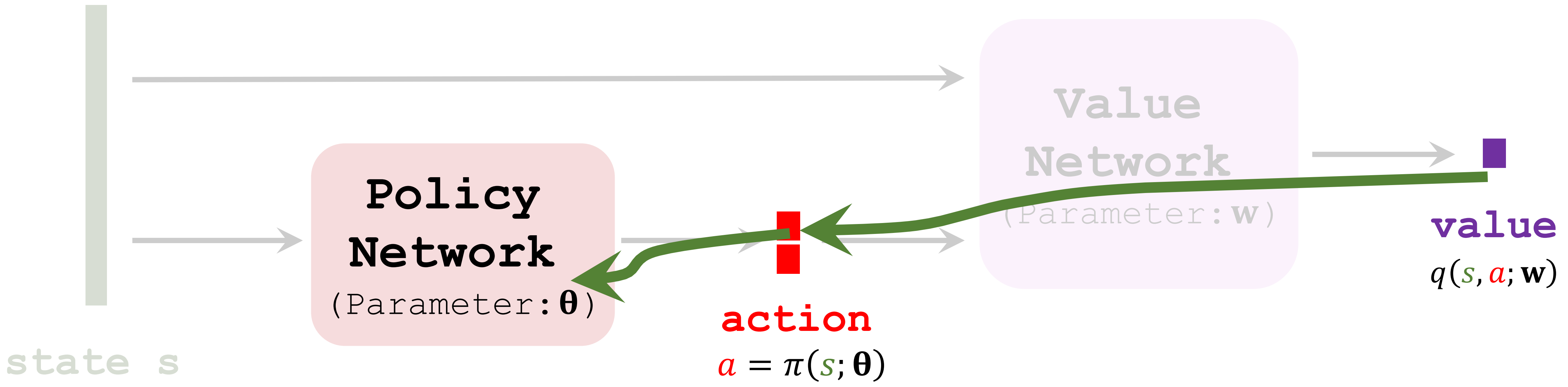
Updating Policy Network by DPG

- **Goal:** Increasing $q(s, a; \mathbf{w})$, where $a = \pi(s; \boldsymbol{\theta})$.
- DPG: $\mathbf{g} = \frac{\partial\, q(s,\pi(s;\boldsymbol{\theta});\mathbf{w})}{\partial\, \boldsymbol{\theta}} = \frac{\partial\, a}{\partial\, \boldsymbol{\theta}} \cdot \frac{\partial\, q(s,a;\mathbf{w})}{\partial\, a}$.



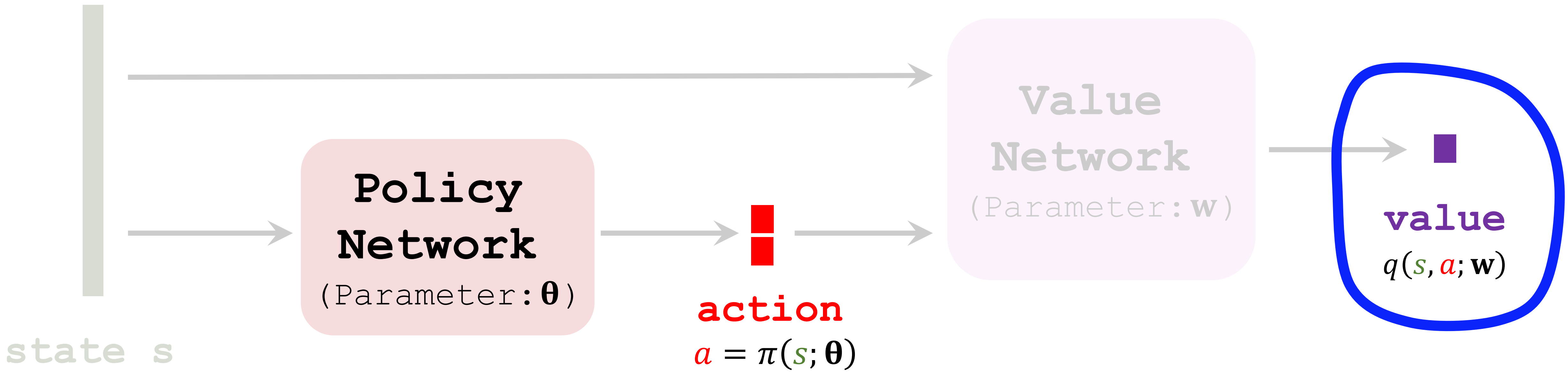
Updating Policy Network by DPG

- **Goal:** Increasing $q(s, a; \mathbf{w})$, where $a = \pi(s; \boldsymbol{\theta})$.
- DPG: $\mathbf{g} = \frac{\partial\, q(s,\pi(s;\boldsymbol{\theta});\mathbf{w})}{\partial\, \boldsymbol{\theta}} = \frac{\partial\, a}{\partial\, \boldsymbol{\theta}} \cdot \frac{\partial\, q(s,a;\mathbf{w})}{\partial\, a}$.



Updating Policy Network by DPG

- **Goal:** Increasing $q(s, a; \mathbf{w})$, where $a = \pi(s; \boldsymbol{\theta})$.
- DPG: $\mathbf{g} = \frac{\partial\, q(s,\pi(s;\boldsymbol{\theta});\mathbf{w})}{\partial\, \boldsymbol{\theta}} = \frac{\partial\, a}{\partial\, \boldsymbol{\theta}} \cdot \frac{\partial\, q(s,a;\mathbf{w})}{\partial\, a}$.
- Gradient ascent: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \beta \cdot \mathbf{g}$.



Stochastic Policy VS Deterministic Policy

	Stochastic Policy	Deterministic Policy
Policy:	$\pi(a \mid s; \boldsymbol{\theta})$	$\pi(s; \boldsymbol{\theta})$

	Stochastic Policy	Deterministic Policy
Policy:	$\pi(a \mid s; \boldsymbol{\theta})$	$\pi(s; \boldsymbol{\theta})$
Output:	Probability distribution over the action space	Action a

	Stochastic Policy	Deterministic Policy
Policy:	$\pi(a \mid s; \boldsymbol{\theta})$	$\pi(s; \boldsymbol{\theta})$
Output:	Probability distribution over the action space	Action a
Control:	Randomly sample an action from the distribution	Directly use the output, a

	Stochastic Policy	Deterministic Policy
Policy:	$\pi(a \| s; \boldsymbol{\theta})$	$\pi(s; \boldsymbol{\theta})$
Output:	Probability distribution over the action space	Action a
Control:	Randomly sample an action from the distribution	Directly use the output, a
Application:	Mostly discrete control	Continuous control