# Policy gradient methods

# Plan

1. Recap.
2. Quiz
3. Q-learning problems
4. Policy methods
5. Logderivative trick
6. REINFORCE
7. Actor-Critic
8. Advantage Actor-Critic

# Recap. RL Notation

| | |
|---|---|
| | Action |
| | State |
| | Reward |
| $\pi(s)$ | Policy |
| $V_\pi(s)$ | Value |
| $Q_\pi(s,a)$ | Q-Value |

# Recap. RL Notation

| | | |
|---|---|---|
| | Action | all possible moves |
| | State | current situation |
| | Reward | immediate return |
| $\pi(s)$ | Policy | strategy determines next action |
| $V_\pi(s)$ | Value | expected long-term return |
| $Q_\pi(s,a)$ | Q-Value | expected long-term return |

# Recap

- **Value based**

- **Policy based**

# Recap

- **Value based**

Learn value function $Q_\theta(s,a) \quad V_\theta(s)$

Infer policy $\pi(a|s) = \mathrm{argmax}_a Q_\theta(s,a)$

- **Policy based**

Explicitly learn policy $\pi_\theta(a|s)$

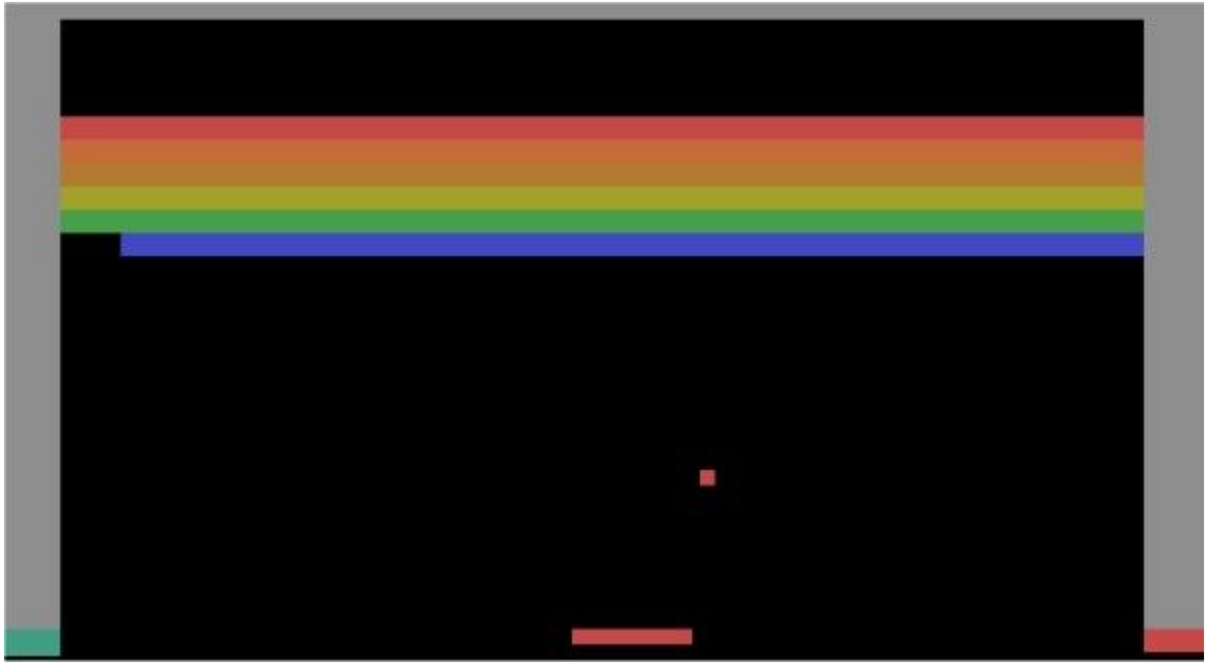
Implicitly maximize reward

# Quiz

The next slide contains a question.

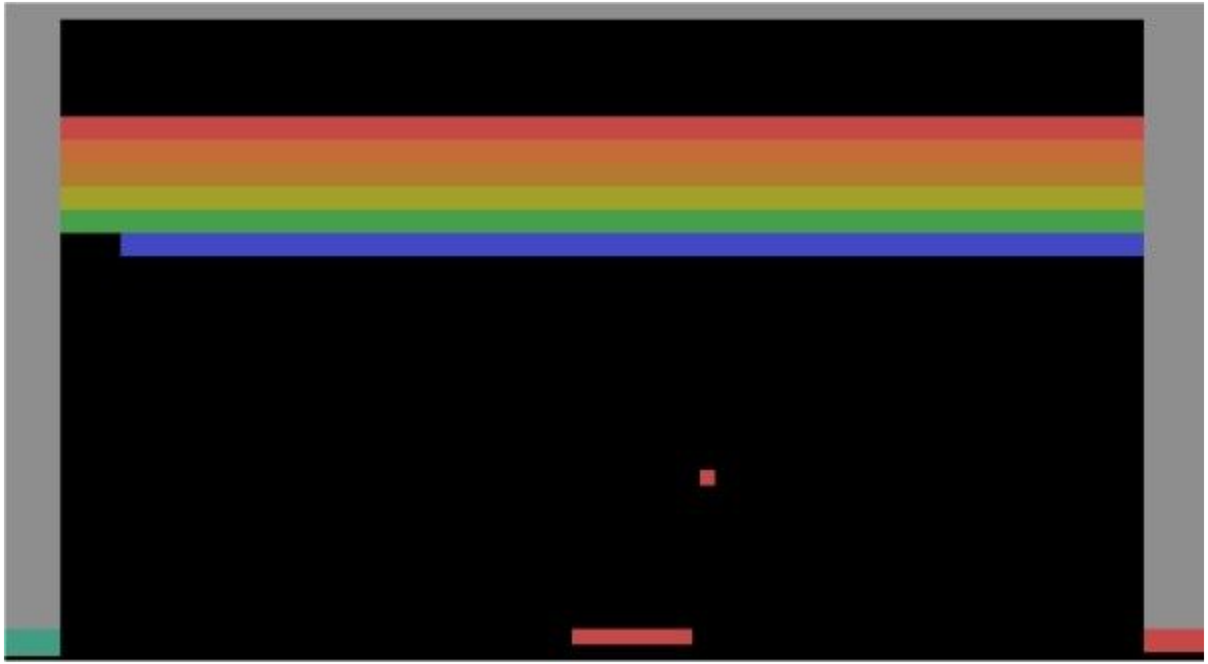
Please respond as fast as you can!

# Quiz



**Left** or **right**?

# Quiz



What's **Q(s, right)** under gamma=0.99?

# Approximation error

DQN objective: $$L \approx E[Q(s_t, a_t) - (r_t + \gamma \max_{a'} Q(s_{t+1}, a'))]^2$$

Simple 2-state world

| | True | (A) | (B) |
|---|---|---|---|
| $Q(s_0, a_0)$ | 1 | 1 | 2 |
| $Q(s_0, a_1)$ | 2 | 2 | 1 |
| $Q(s_1, a_0)$ | 3 | 3 | 3 |
| $Q(s_1, a_1)$ | 100 | 50 | 100 |

Which works better?

# Approximation error

DQN objective: $$L \approx E[Q(s_t, a_t) - (r_t + \gamma \max_{a'} Q(s_{t+1}, a'))]^2$$

Simple 2-state world

| | True | (A) | (B) |
|---|---|---|---|
| $Q(s_0, a_0)$ | 1 | 1 | 2 |
| $Q(s_0, a_1)$ | 2 | 2 | 1 |
| $Q(s_1, a_0)$ | 3 | 3 | 3 |
| $Q(s_1, a_1)$ | 100 | 50 | 100 |

**Better policy** **Less MSE**

# Conclusion

- Q-values could be harder to compute than to pick optimal action
- We could avoid learning value functions by directly learning agent's policy $\pi_\theta(a|s)$

# Policies

- Deterministic policy

$$a = \pi_\theta(s)$$

- Stochastic policy

$$a \sim \pi_\theta(a|s)$$

# Policies

- Deterministic policy

$$a = \pi_\theta(s)$$

- Stochastic policy

$$a \sim \pi_\theta(a|s)$$

**When stochastic is better?**

# Policies

- Deterministic policy

$$a = \pi_\theta(s)$$

- Stochastic policy

$$a \sim \pi_\theta(a|s)$$

**When stochastic is better?**

**E.g. rock-paper-scissors**

# Policies

- Deterministic policy

$$a = \pi_\theta(s)$$

- Stochastic policy

$$a \sim \pi_\theta(a|s)$$

Other pros?

# Policies

- Deterministic policy

$$a = \pi_\theta(s)$$

- Stochastic policy

$$a \sim \pi_\theta(a|s)$$

**Other pros?**

- exploration
- continuous action space

# Recap. Crossentropy method

- Initialize policy params

- Loop:
  - sample N sessions
  - elite = take M best sessions and concatenate

$$\theta_{i+1} = \theta_i + \alpha \nabla \sum_k \log \pi_\theta(a_k | s_k) \; [s_k, a_k \in \text{elite}]$$

# Recap. Crossentropy method

- Initialize policy params

- Loop:
  - sample N sessions
  - elite = take M best sessions and concatenate

$$\theta_{i+1} = \theta_i + \alpha \nabla \sum_k \log \pi_\theta(a_k | s_k) \; [s_k, a_k \in \text{elite}]$$

What about maximization over policy?

# Objective

Expected reward

$$J = E_{s \sim p(s), a \sim \pi_\theta(a|s)} R(s, a, s', a', \ldots)$$

Expected discounted reward

$$J = E_{s \sim p(s), a \sim \pi_\theta(a|s)} Q(s, a)$$

# Objective

$$J = EQ(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a)\, da\, ds$$

# Objective

Agent's policy

$$J = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a)\, da\, ds$$

State visitation frequency

True action value

# Objective

Agent's policy

$$J = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a) \, da \, ds$$

State visitation frequency

True action value

How to compute this?

# Objective

$$J = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a) \, da \, ds$$

Sample N sessions

$$J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a) \in z_i} Q(s,a)$$

# Objective

$$J = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a)\, da\, ds$$

Sample N sessions

$$J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a) \in z_i} Q(s,a)$$

How to compute? $\frac{\partial J}{\partial \theta}$

# Optimization

What about finite difference?

- Change policy a little, evaluate

$$\nabla J \approx \frac{J_{\theta+\epsilon} - J_{\theta}}{\epsilon}$$

# Optimization

$$J = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a) \, da \, ds$$

Whish list:

- Analytical gradient
- Easy approximations

# Logderivative trick

$$\nabla \log \pi(z) = \frac{1}{\pi(z)} \nabla \pi(z)$$

$$\pi(z) \nabla \log \pi(z) = \nabla \pi(z)$$

# Optimization

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a)\, da\, ds$$

# Optimization

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a)\, da\, ds$$

$$\pi(z) \nabla \log \pi(z) = \nabla \pi(z)$$

# Optimization

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a)\, da\, ds$$

$$\pi(z) \nabla \log \pi(z) = \nabla \pi(z)$$

$$\nabla J = \int_s p(s) \int_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) Q(s,a)\, da\, ds$$

# Policy gradient. REINFORCE

- Policy gradient

$$\nabla J = E_{s\sim p(s),a\sim\pi_\theta(a|s)} \nabla \log \pi_\theta(a|s) Q(s,a)$$

- Approximate with sampling

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a)\in z_i} \nabla \log \pi_\theta(a|s) Q(s,a)$$

# REINFORCE baseline

- Initialize NN weights
- Loop:

 - Sample N sessions **z** under current $\pi_\theta(a|s)$
 - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a) \in z_i} \nabla \log \pi_\theta(a|s) Q(s,a)$$

- Ascend

$$\theta_{i+1} = \theta_i + \alpha \nabla J$$

# REINFORCE

- Initialize NN weights
- Loop:

 - Sample N sessions **z** under current $\pi_\theta(a|s)$
 - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a)\in z_i} \nabla \log \pi_\theta(a|s) Q(s,a)$$

$$Q(s,a) = V(s) + A(s,a)$$

# REINFORCE

- Initialize NN weights
- Loop:

 - Sample N sessions **z** under current $\pi_\theta(a|s)$
 - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N}\sum_{i=0}^{N}\sum_{(s,a)\in z_i}\nabla \log \pi_\theta(a|s)\left(Q(s,a)-b(s)\right)$$

# Actor-critic

- Learn both $V(s)$ and $\pi_\theta(a|s)$

# Advantage Actor-critic

- Idea: learn both $V_\theta(s)$ and $\pi_\theta(a|s)$
- Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster

How can we estimate $A(s,a)$ from $(s,a,r,s')$ and V-function?

# Advantage Actor-critic

$$A(s,a) = Q(s,a) - V(s)$$

$$Q(s,a) = r + \gamma V(s')$$

$$A(s,a) = r + \gamma V(s') - V(s)$$

# Advantage Actor-critic

- Idea: learn both $V_\theta(s)$ and $\pi_\theta(a|s)$
- Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster

$$A(s,a) = r + \gamma V(s') - V(s)$$

$$\nabla J_{actor} \approx \frac{1}{N}\sum_{i=0}^{N}\sum_{(s,a)\in z_i} \nabla \log \pi_\theta(a|s) A(s,a)$$

# Advantage Actor-critic

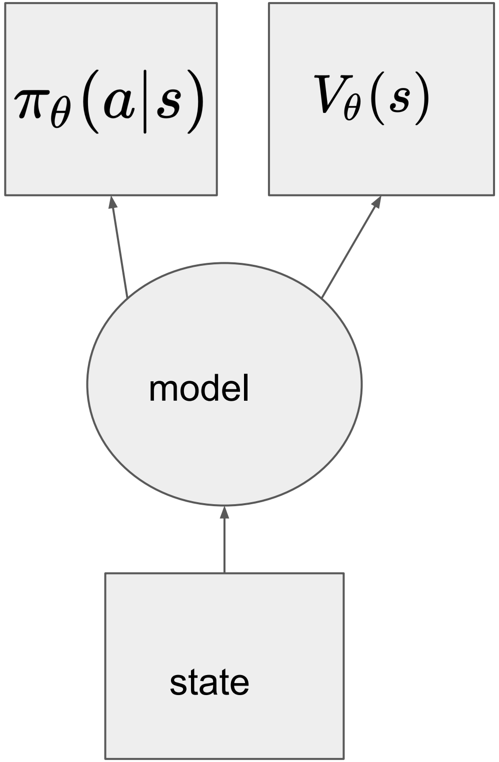
- Idea: learn both $V_\theta(s)$ and $\pi_\theta(a|s)$
- Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster

**How to learn V(s)?**

$$A(s,a) = r + \gamma V(s') - V(s)$$

$$\nabla J_{actor} \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a)\in z_i} \nabla \log \pi_\theta(a|s) A(s,a)$$

# Advantage Actor-critic algorithm




Improve policy

$$\nabla J_{actor} \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a)\in z_i} \nabla \log \pi_\theta(a|s) A(s,a)$$

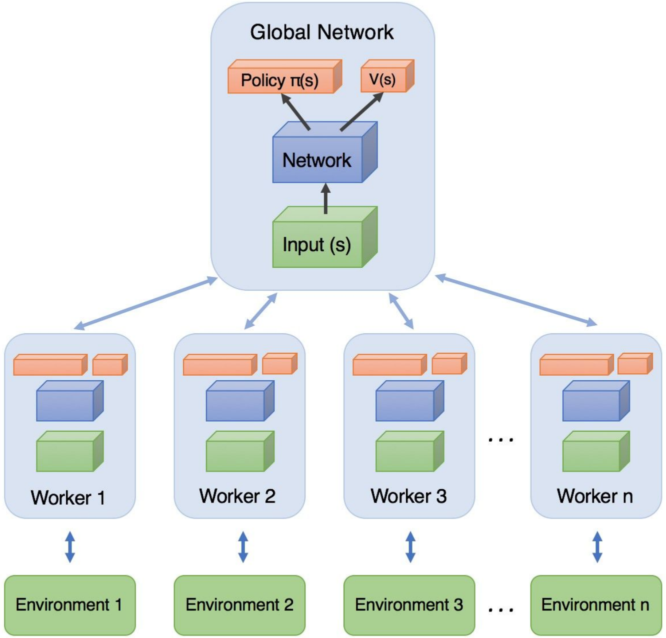
Improve value

$$L_{critic} \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{(s,a)\in z_i} (V_\theta(s) - [r + \gamma V(s')])^2$$

# Asynchronous Advantage Actor-Critic (A3C)

A popular implementation of a2c using neural net agent

- No experience replay
- Many parallel sessions
- Asynchronous updates
- Recurrent memory



# Value-based vs policy-based

- Q-learning, SARSA

- Solves harder problem
- Explicit exploration
- Evaluates states and actions
- Easier to train off-policy

- REINFORCE, A2C

- Solves easier problem
- Innate exploration and stochasticity
- Easier for continuous actions
- Compatible with supervised learning