4/16/2024: Deep Q-Learning, Policy gradient

3rd Q-Learning variant: Deep Q-Learning

Idea: $\hat{Q}(s,a)$ = output of a neural network whose input is s & a

Let θ be parameters of model.

$$\text{Loss}(\theta) = \frac{1}{2}\Big(\underbrace{\hat{Q}_\theta(s,a)}_{\text{prediction from NN with parameters } \theta} - \underbrace{[r + \gamma \hat{V}(s')]}_{\text{"target"}}\Big)^2$$

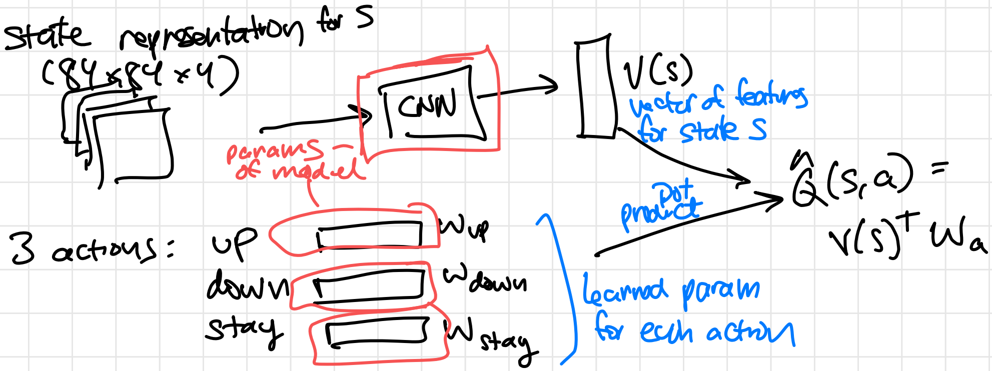
minimize squared difference

$$\nabla_\theta \text{loss}(\theta) = \cancel{\tfrac{1}{2}} \cdot \cancel{2} \cdot \underbrace{\big(\hat{Q}_\theta(s,a) - [r + \gamma \hat{V}(s')]\big)}_{\text{compute directly}} \cdot \underbrace{\nabla_\theta \hat{Q}_\theta(s,a)}_{\text{computed by backprop}}$$

Update θ with gradient descent when we receive new rewards r & new state s'

Example DQN for Pong

Represent state as last k frames
- Each frame is 84×84
- Set $k = 4$
- Input is $84 \times 84 \times 4$ block of numbers



Policy Gradient methods

- Q-Learning: Given (s,a), predict $Q(s,a) \in \mathbb{R}$
  "Regression"

- Policy Gradient: Given s, predict optimal a
  "classification"

Input State representation $\longrightarrow$ $\boxed{\text{Model } \theta}$ $\longrightarrow$ probability distribution over actions

$$\underbrace{\pi_\theta(a \mid s)}_{\text{probabilistic policy}}$$

How to train?

- Can't do supervised learning b/c no supervised training data
- Solution: Train $\pi_\theta(a|s)$ to achieve large expected total reward

Want to maximize:

$$\underbrace{V(\theta)}_{\text{Expected sum of rewards when using policy } \pi_\theta(a|s)} = \sum_{\text{trajectories } z} \underbrace{P(z;\theta)}_{\text{Prob. that } z \text{ happens}} \cdot \underbrace{R(z)}_{\text{total reward of } z} $$

$$R(z) = \sum_{t=1}^{T} r_t$$

z: all possible sequences $[s_1, a_1, r_1, s_2, a_2, r_2, s_3 \cdots]$

Plan: maximize $V(\theta)$, use gradient ascent.
Need to compute $\nabla_\theta V(\theta)$

Useful trick:

$$\nabla_\theta \log P(z;\theta) = \frac{1}{P(z;\theta)} \nabla_\theta P(z;\theta)$$

$$\Longleftrightarrow \quad \nabla_\theta P(z;\theta) = P(z;\theta) \cdot \nabla_\theta \log P(z;\theta)$$

$$\nabla_\theta V(\theta) = \underbrace{\sum_{\substack{\text{traj's} \\ z}} P(z;\theta)}_{\text{expected value ... of}} \cdot \underbrace{\nabla \log P(z;\theta) \cdot R(z)}_{\text{this}}$$

$$= \mathbb{E}_{z \sim P_\theta} \left[\boxed{\nabla \log P(z;\theta)} \cdot R(z) \right]$$

Approximate this with sampled trajectories & computing the mean

computing this

$$\log P(z;\theta) = \boxed{\log P(s_1)} + \boxed{\log \pi(a_1 | s_1)} + \boxed{\log T(s_1, a_1, s_2)}$$
$$+ \boxed{\log \pi(a_2 | s_2)} + \boxed{\log T(s_2, a_2, s_3)}$$
$$+ \dots$$

✗ unknown to us

∵ does not depend on θ

so ∇_θ is 0

compute ∇_θ with backprop